НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:
Завідувач кафедри
_____ *Сергій СТІРЕНКО*
(підпис)
“___”_____20__ р

Дипломний проект

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерні системи та мережі»

спеціальності 123 «Комп’ютерна інженерія»

на тему: «Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації»

Виконала:
студентка _IV_курсу, групи__ІО-64__

____Буровецька Ксенія Олександрівна_____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник _доцент Марковський Олександр Петрович_ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Консультант __н. контроль_ _доц. д.т.н. Сімоненко В.П._ _____
(назва розділу) (вчені ступінь та звання, прізвище, ініціали) (підпис)

Рецензент _доц. каф. СПіСКС к.т.н.,доц Марія ОРЛОВА_ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМ. ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 123 «Комп'ютерна інженерія»
Освітньо-професійна програма «Комп'ютерні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ *Сергій СТІРЕНКО*
(підпис)
"___"_____20__ р.

ЗАВДАННЯ

на дипломний проект студенту

Буровецькій Ксенії Олександрівні

1. Тема проекту «*Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації»*

керівник проекту Марковський Олександр Петрович, доцент, затверджені наказом по університету від «*07*» *травня* 2020р. №1081-с

2. Термін здачі студентом закінченої роботи *2020р.*

3. Вихідні дані до проекту *технічне завдання, теоретичні дані.*

4. Зміст пояснювальної записки: *опис предметної області, дослідження методів та програмних засобів побудови булевих перетворень для криптографічних алгоритмів захисту інформації, програма, що генерує SAC-функцій за методом розкладання на систему балансних підфункцій.*

5. Консультант роботи, з вказівкою розділів роботи, які до них вносяться

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Нормоконтроль	Сімоненко В.П.		

6. Дата видачі завдання *01.09 .2019 року*

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту(роботи)	Примітки
1.	*Затвердження теми роботи*	01.09.2019	
2.	*Вивчення та аналіз завдання*	10.09.2019	
3.	*Розробка структури програми*	01.10.2019	
4.	*Програмна реалізація*	13.02.2020	
5.	*Оформлення пояснювальної записки*	02.04.2020	
6.	*Передзахист*	26.05.2020	
7.	*Захист*	25.06.2020	

Студент Ксенія БУРОВЕЦЬКА _____

(підпис)

Керівник Олександр МАРКОВСЬКИЙ _____

(підпис)

Анотація

В бакалаврському дипломному проекті розроблено програму, що генерує SAC-функції за методом розкладання на систему балансних підфункцій, що призначена для отримання алгебраїчної нормальної форми булевих функцій, що задовольняють критерію строгого лавинного ефекту, тобто SAC-критерію.

Перевагою запропонованої методики побудови SAC-функцій є простота і істотно менші витрати машинного часу в порівнянні з відомими методами вирішення цього завдання.

Annotation

In this bachelor's diploma project was implemented a program that generates SAC-functions by the method of decomposition into a system of balanced subfunctions, which is designed to obtain an algebraic normal form of Boolean functions that satisfy the Strict Avaluate Criterion, ie SAC-criterion.

The advantage of the proposed method of constructing SAC-functions is the simplicity and significantly lower cost of machine time compared to known methods for solving this problem.

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проект	2	
2	А4	ІАЛЦ.467200.001 ВП	Відомість проекту	1	
3	А4	ІАЛЦ.467200.002 ТЗ	Технічне завдання	4	
4	А4	ІАЛЦ.467200.003 ПЗ	Пояснювальна записка	59	
5	А4	ІАЛЦ.467200.004 Д1	Принципова схема	1	
6	А4	ІАЛЦ.467200.005 Д2	Структурна схема	1	
7	А4	ІАЛЦ.467200.006 Д3	Функціональна схема	1	
8	А4	ІАЛЦ.467200.007 Д4	Лістинг програми синтезу булевих функцій	8	

					ІАЛЦ.467200.001 ВП	
Зм.		*№ документа*	*Підпис*	*Дата*		
Розробив		*Буровецька К.О*			Відомість дипломного проекту	*Літ.* / *Аркуш* / *Аркушів*
Перевірив		*Марковський О.П.*				*1* / *1*
Н. Контр.		*Сімоненко В.П.*				*НТУУ «КПІ ім. Ігоря Сікорського» ФІОТ гр. ІО-64*
Затвердив		*Стіренко* С.Г.				

ТЕХНІЧНЕ ЗАВДАННЯ

до дипломної роботи
освітньо-кваліфікаційного рівня___бакалавр___

на тему: “ Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації”

Київ – 2020 року

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Найменування: «Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації».

Область застосування: для синтезу та аналізу функціональних перетворень для алгоритмів криптографічного захисту.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання бакалаврського дипломного проекту, затверджене кафедрою обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут».

3. МЕТА І ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою розробки є створення апаратно-програмного комплексу для синтезу та аналізу функцій спеціальних класів.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки є науково-технічна література по вивченню функцій спеціальних класів та особливості їх синтезу, публікації в періодичних виданнях, довідники по платформах дистанційного навчання, публікації в Інтернеті з даних питань.

ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до розробляємого продукту

Додаток, що розробляється повинен:

- Виконувати синтез функціональних перетворень для алгоритмів криптографічного захисту;
- Інформативний показ етапів синтезу функцій спеціальних класів;
- Синтез функціональних перетворень для алгоритмів криптографічного захисту;
- Можливість запуску програми на великій кількості пристроїв;
- Можливість забезпечення комфортної роботи для користувачів з різною роздільною здатністю екрану.

5.2. Вимоги до програмного забезпечення

- Операційна система Windows7 та вище;
- Наявність браузеру версій випуску не раніше 2010 року.

5.3. Вимоги до апаратної частини

- Пристрій з двохядерним процесором з частотою 1,5 ГГц;
- Екран з діагоналлю від 5 дюймів;
- Оперативна пам’ять 1 Гб.

1. ЕТАПИ РОЗРОБКИ

	Дата
Вивчення літератури	27.11.2019
Складання і узгодження технічного завдання	20.01.2019
Написання вступної частини та огляд рішень	15.03.2020
Створення модулів розроблюваного продукту	16.04.2020
Тестування продукту та корекція помилок	16.05.2020
Оформлення документації дипломного проекту	20.05.2020
Подання ДП рецензенту	03.06.2020
Захист дипломного проекту	25.06.2020

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту
на тему: «Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації»

Київ – 2020 року

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

SAC (Strict Avaluate Criterion) - критерій строгого лавинного ефекту.
НВІС (VLSI) - надвелика інтегральна схема.
АНФ – алгебраїчна нормальна форма.

ВСТУП

Основним стимулом розвитку систем захисту інформації та комп'ютерної безпеки, яка, як частина комп'ютерної інженерії інтенсивно розвивається в останні роки стало широке впровадження мережевих технологій і інтегрованих систем обробки даних.

Розвиток мережевих технологій обробки даних дозволив досягти якісно нових показників систем обробки інформації, багаторазово розширив сферу використання засобів обчислювальної техніки і засобів електронної обробки даних. Поява таких технологій докорінно змінила і концептуальну основу побудови баз даних, істотно підвищивши можливості накопичення і поновлення даних, зробивши їх інтегрованими. Одним з найважливіших додатків мережевих технологій стало їх використання в сфері бізнесу та банківській справі. Саме мережі забезпечують високу гнучкість, оперативність та надійність реалізації інформаційних потоків в цих сферах. Важливою стороною для згаданих додатків є забезпечення конфіденційності та достовірності передачі інформації по мережах. В тій чи іншій мірі сказане стосується використання мережевих технологій в інших сферах. Таким чином, необхідною умовою подальшого розвитку і впровадження мережевих технологій у всіх галузях людської діяльності стало забезпечення надійного захисту інформації в мережах і інтегрованих базах даних. При проходженні інформаційних повідомлень по мережах існує потенційна небезпека їх несанкціонованого читання, зміни і підміни. Значну небезпеку представляє також можливість несанкціонованого проникнення в інтегровані бази даних і системи обробки інформації. Саме цими причинами і обумовлений різко збільшений в останні роки теоретичний і практичний інтерес до систем захисту інформації.

Слід особливо відзначити в корені змінений характер систем захисту інформації. Сучасні системи захисту інформації є комплексом організаційних, алгоритмічних і технічних засобів, спрямованих на вирішення трьох основних завдань:

- забезпечення надійного захисту інформації від несанкціонованого доступу, тобто читання при передачі по мережах або при зберіганні в пам'яті інтегрованих систем;
- забезпечення захисту автентичності інформації, переданої по мережах і інформації, що зберігається в пам'яті інтегрованих баз даних, тобто її захист від несанкціонованих змін;
- забезпечення аутентифікації абонентів мережі і користувачів інтегрованих систем обробки інформації, тобто захист від несанкціонованого доступу в інтегровані системи і бази даних.

Добре відомо, що проблема захисту інформації виникла практично з виникненням писемності. Однак до використання сучасних мережевих технологій, засоби захисту використовувалися для вирішення тільки перших двох з перерахованих вище завдань. Криптографічне шифрування широко застосовувалося починаючи з античних часів у всіх країнах світу. Практично до кінця 70-х років методи і засоби захисту інформації відносились до закритої області комп'ютерних технологій. Така ситуація зберігалася за умови дуже обмеженого кола споживачів методів і засобів захисту інформації, які відносились майже виключно до державних організацій. Це, з іншого боку, дозволяло забезпечувати досить кваліфіковане використання засобів захисту інформації.

З розвитком мережевих технологій багатократно розширилося коло користувачів систем захисту інформації, різко знизилася їхня кваліфікація в сфері захисту інформації, виникли нові завдання захисту інформації і це зумовило нові підходи для її забезпечення, які можна сформулювати у вигляді наступних принципів:

- забезпечення криптографічного захисту інформації повинно грунтуватися не на закритості криптографічних алгоритмів, а на математичних принципах, покладених в їх основу;

- створення, сертифікація та впровадження протоколів захисту інформації та алгоритмів має виконуватися спеціальними міжнародними організаціями та самі вони повинні зводитися в ранг юридично оформленого стандарту;

- рівень захисту даних, що забезпечується криптографічним алгоритмом повинен визначатися особливостями застосування і бути таким, щоб витрати на розкриття були більші потенційного прибутку, який може бути отриманий від розкриття алгоритмів.

У розвитку першого принципу необхідно відзначити, що в основі всіх сучасних алгоритмів криптографічного захисту інформації лежить, так звана, нерозв'язна математична задача. Зокрема, в основі широко відомого алгоритму RSA лежить задача факторизації великих чисел, в основі іншого алгоритму - ElGamal лежить нерозв'язна аналітично задача дискретного логарифмування. Для захисту інформаційних файлів в базах даних особливу роль грають симетричні алгоритми шифрування і хеш-алгоритми, в основі криптографічних властивостей яких лежить нерозв'язна задача знаходження коренів нелінійних булевих рівнянь. В ході поступального розвитку теорії та практики захисту інформації вдосконалюються всі її елементи, виходячи з нових технічних можливостей мікроелектроніки, обчислювальної техніки і засобів передачі даних. На динаміку розвитку засобів захисту інформації активний вплив має вдосконалення методів і засобів порушення безпеки. Основним елементом більшості сучасних систем захисту інформації є її криптографічне перетворення. Швидке зростання продуктивності сучасних засобів обчислювальної техніки і можливості організації паралельного вирішення завдань криптоаналізу на комп'ютерах, об'єднаних в мережі, різко знижують ефективність традиційних криптографічних алгоритмів. Крім того,

спільна відкрита участь в дослідженнях з проблеми криптографічного захисту інформації великого числа вчених з різних країн світу дозволила в останні роки отримати значні результати в області теоретичного криптоаналізу, що вимагає істотного перегляду існуючих і розробки нових криптографічних алгоритмів виходячи з нових концепцій криптографічного захисту інформації. Одним з напрямків розвитку нових концепцій захисту інформації, інтерес до якого, якщо судити за кількістю публікацій, помітно зріс - є функціональний аналіз процедур перетворення інформації на рівні булевих функцій. Вказане зростання практичного інтересу до концепції дослідження криптографічних алгоритмів на рівні булевих функцій обумовлене в першу чергу прогресом обчислювальної техніки, багаторазовий ріст продуктивності якої і збільшення обсягів пам'яті надають можливість уже на сучасному етапі виконувати досить трудомісткий аналіз складних алгоритмів на рівні бітових перетворень. Це має велике практичне значення як для аналізу існуючих алгоритмів криптографічного перетворення, так і при розробці нових алгоритмів. Це вимагає розробки нових методів отримання булевих функціональних перетворень, орієнтованих на використання в системах криптографічного захисту інформації та перетворень , що володіють специфічними властивостями. Таким чином, тема бакалаврської роботи, спрямованої на створення нових методів отримання булевих функцій спеціальних класів для алгоритмів криптографічного захисту інформації є актуальною.

РОЗДІЛ 1.
АНАЛІЗ ВИКОРИСТАННЯ В КРИПТОГРАФІЧНИХ АЛГОРИТМАХ БУЛЕВИХ ПЕРЕТВОРЕНЬ І ОГЛЯД МЕТОДІВ ЇХ ОТРИМАННЯ

1.1. Особливості захисту інформації в базах даних і аналіз факторів, що впливають на рівень захищеності

Розвиток мережевих технологій привів до широкого поширення інтегрованих баз даних, включених в локальну або глобальну мережу. Зберігання інформації в сучасних базах даних може реалізуватися або на спеціальному комп'ютері, підключеному до потужної системі дискової магнітної і оптичної пам'яті, або з рознесенням інформації по окремих комп'ютерах, підключених, найчастіше, в загальну локальну мережу [1]. Користувачі такої інтегрованої бази даних можуть виконувати операції пошуку і отримання інформації, або можуть вносити контрольовані зміни.

Система захисту інформації баз даних покликана вирішувати такі завдання, наведені в порядку їх важливості:

- аутентифікація користувачів інтегрованої бази даних з тим, щоб визначити права їх доступу, при цьому слід врахувати, що фактично доступ частіше віддалений, тобто по локальній або глобальній комп'ютерній мережі.
- захист даних від несанкціонованого читання користувачами, які не мають допуску до певних файлів.
- захист автентичності інформаційних файлів і захист від внесення користувачами несанкціонованих змін в базу даних;

Для вирішення цих завдань використовуються всі типи криптографічних алгоритмів. Найбільш часто використовуються хеш-алгоритми, що формують хеш-сигнатури інформаційних файлів, що зберігаються на диску. Важливим застосуванням хеш-сигнатур є підтвердження автентичності інформації, що зберігається в базах даних, до рівня захищеності яких пред'являються підвищені вимоги, наприклад бази даних банків [2]. Протокол передбачає наявність в системі закритої по запису пам'яті хеш-сигнатур, запис до якої може виконуватися тільки відповідальною особою (керуючим банку) і в даному разі можна вважати, що запис хеш-сигнатури інформаційного блоку еквівалентний підпису керуючого під інформацією, що в ньому міститься.

В процесі роботи запису інформації, вона записується в дискову пам'ять, доступну для запису для обмеженого кола осіб і доступну для зчитування для більш широкого кола користувачів, для яких доступною для читання є і інформація, що зберігається в закритій пам'яті хеш-сигнатур. При читанні інформації з основної дискової пам'яті бази даних виконується обчислення за відомим відкритим алгоритмом коду хеш-сигнатури, який порівнюється з кодом хеш-сигнатури, що зберігається в закритій пам'яті: збіг зазначених кодів свідчить про справжність зчитаної з основної пам'яті інформації [3, 4].

При запису інформації копія її надходить на комп'ютер відповідальної особи, де відбувається контроль цієї інформації і обчислення з відкритого алгоритму її хеш-сигнатури, яка записується в закриту пам'ять хеш-сигнатур. Таким чином, в описаному вище протоколі захисту інформації в базах даних хеш-сигнатури використовуються для контролю автентичності інформаційних повідомлень.

Найважливішою сферою використання цифрових сигнатур і хеш-сигнатур в сучасних системах захисту інформації є аутентифікація користувачів в обчислювальних системах і інтегрованих базах даних [5].

У більшості сучасних систем використовуються паролі. Основні вимоги до паролю зводяться до наступного: код пароля, по можливості, повинен бути якомога менш передбачуваним, мати якомога більшу довжину, але в той же час, повинен бути стійкий до забування користувачем [6]. Найбільш доцільним є варіант, при якому код пароля генерується самим користувачем - це повністю виключає небезпеку пов'язану з тим, що пароль знають кілька осіб.

Зберігання в пам'яті паролів користувачів з процедурою порівняння введеного пароля зі списком паролів - найбільш "логічне" рішення проблеми аутентифікації в більшості випадків виявляється неприйнятним, так як при цьому потрібно надійно захистити пам'ять, де зберігається список паролів для читання і запису. Більш ефективним є протокол ідентифікації користувачів, в основі якого лежить зберігання в пам'яті не паролів в явному вигляді, а їх хеш-сигнатур (або сигнатур) [7]. При цьому код пароля присутній в обчислювальній системі тільки на етапі введення і тут же перетворюється в хеш-сигнатуру, яка порівнюється зі списком хеш-сигнатур, що зберігаються в пам'яті, яка повинна тепер бути закритою тільки для операції записи і може бути відкритою для читання. Описаний протокол, в якому використовуються хеш-сигнатури дійсних паролів забезпечує можливість того, що користувач сам генерує пароль і в спеціальному режимі повідомляє обчислювальної системі тільки сигнатуру обраного пароля [8].

Для забезпечення захисту інформації від несанкціонованого читання в інтегрованих базах даних широко використовуються симетричні алгоритми шифрування, які забезпечують суттєво більшу швидкодію (приблизно на 3 порядки) в порівнянні з алгоритмами з відкритим ключем [9, 10].

Найбільш поширеним засобом захисту інформації є її функціональне перетворення з метою унеможливити доступ до неї без знання ключа (криптографічне кодування) або з метою підтвердження її достовірності (цифрові хеш-сигнатури).

Алгоритми функціонального перетворення в системах захисту інформації, що розглядаються в загальному вигляді, повинні відповідати двом основним вимогам: з одного боку вони повинні забезпечувати високий рівень захищеності - тобто володіти достатньою складністю, щоб унеможливити практичне виконання зворотного перетворення (отримання ключа по вихідному і перетвореному повідомленням в разі криптографічного кодування або отримання повідомлення-синоніма за відомим кодом хеш-сигнатури) [11], а з іншого - бути достатньо простими з тим, щоб їх реалізація не позначалася помітним чином на швидкості обробки інформації. Багаторазово збільшені в останні роки можливості апаратних засобів обчислювальної техніки змушують переглянути сформовані концепції побудови алгоритмів функціональних перетворень в системах захисту інформації [12].

В узагальненому вигляді процес функціонального перетворення інформації в системах її захисту можна формально розглядати як певну функцію відображення блоку інформаційного повідомлення і ключа в вихідний код (закодоване повідомлення або цифрову хеш-сигнатуру). Оскільки практично всі алгоритми перетворення інформації в системах її захисту припускають розбиття вихідного інформаційного повідомлення на блоки фіксованої довжини l, то з достатнім ступенем загальності можна говорити про те, що функція перетворення двійкового l-мірного вектора M блоку інформаційного повідомлення являє собою систему булевих функцій $g_i(M, K)$, $i = 1,...,d$, (причому $d = l$ для систем криптографічного кодування), що утворюють безліч G і певних на безлічі значень l- мірного двійкового вектора M і h-мірного двійкового вектора ключа K [13, 14]. Надалі будемо вважати, що кожна з функцій $g_i(M, K)$ визначена на безлічі наборів, утвореного усіма можливими наборами значень булевих змінних $x_1, ..., x_n$, $n = l + h$. Виходячи з теоретико-інформаційного підходу до процесів криптографічного кодування, суттєвим є те, що для забезпечення стійкості до спроб розкриття, всі ці булеві функції повинні бути по-перше балансними (тобто приймати

одиничні значення на рівно половині з 2^n можливих наборів вхідних змінних) з тим, щоб їх значення-розряди вихідного коду були максимально навантаженими в інформаційному плані і по-друге, бути попарно некорельованими, щоб виключити дублювання інформації, що в них міститься [15].

Таким чином, в основі більшості криптографічних алгоритмів, використовуваних для захисту інформації в інтегрованих базах даних лежить нерозв'язна аналітично задача знаходження коренів системи нелінійних булевих рівнянь. Єдиним способом вирішення цього завдання є перебір. Існує ряд способів зменшення об'єму перебору при вирішенні систем нелінійних булевих рівнянь, які стосовно до задач розкриття отримали назви диференціального і лінійного криптоаналізу. Ефект від використання цих методів розкриття заснований на "слабкостях" булевих функцій, що складають систему еквівалентних криптографічному алгоритму булевих функцій. Показано, що якщо булеві функції, що складають еквівалентну систему мають певні властивості, то зменшення перебору в рамках диференціального і лінійного криптоаналізу не може бути досягнуто і відповідний алгоритм є стійким до розкриття аналітичними методами [16]. Для цього булеві функції повинні відповідати таким вимогам:

1. Кожна з булевих функцій $f_i(x_1,...,x_n,k_1,...,k_r)$, $\forall$ i=1,...,h еквівалентної до криптографічного алгоритму системи, повинна мати властивість максимальної повної ентропії, тобто бути балансною.
2. Кожна з булевих функцій $f_i(x_1,..., x_n, k_1,..., k_r)$, $\forall$ i=1,...,h еквівалентної до криптографічного алгоритму системи, повинна мати максимальну умовну ентропію, тобто часткова функція, що виходить при виключенні будь-якої із змінних $x_1,..., x_n, k_1,..., k_r$ повинна володіти максимумом ентропії, або, що те ж саме бути балансною. Зазначений критерій отримав назву критерію строгого лавинного ефекту (Strict Avaluate Criterion або SAC).

3. Кожна з булевих функцій $f_i(x_1,..., x_n, k_1,..., k_r)$, $\forall$ i=1,...,h еквівалентної до криптографічного алгоритму системи, повинна володіти максимальним значенням нелінійності, тобто бути bent-функцією.
4. Булеві функції, що складають еквівалентну до криптографічного алгоритму систему, повинні бути попарно некорельованими або, що те ж саме, булева функція $\zeta_{ij}(a, x_1,..., x_n, k_1,..., k_r) = a \cdot f_i(x_1,..., x_n, k_1,..., k_r) \oplus a \cdot f_j(x_1,..., x_n, k_1,..., k_r) \oplus f_j(x_1,..., x_n, k_1,..., k_r)$, $\forall$ j,i=1,..h, $i \neq j$ повинна володіти максимумом умовної ентропії по кожній із змінних, на яких вона визначена.
5. Число рівнянь системи має бути максимально великим, тобто якомога більшою повинна бути розрядність блоків, оброблюваних криптографічним алгоритмом.

Необхідно відзначити, що використання еквівалентних до криптографічних алгоритмів систем булевих функцій не обмежується цілями оцінки стійкості до розкриття алгоритмів. Іншим можливим способом використання систем рівнянь для задач криптоаналізу є побудова на їх основі спеціалізованих обчислювальних пристроїв, реалізованих на замовних НВІС, які дозволяють в тисячі разів прискорити виконання криптографічного кодування [17]. Досягнення сучасної інтегральної технології цілком дозволяють реалізувати в рамках однієї НВІС досить складні комбінаційні схеми з використанням регулярних структур. Така НВІС або спеціалізований процесор здатні виконувати завдання розкриття шляхом перебору за досить короткий час - повідомлення про такі розробки зі створення таких НВІС відзначаються в відкритих публікаціях.

Однак дослідження булевих функцій на предмет визначення їх специфічних криптографічних властивостей не обмежується завданнями аналізу. Не менш важливим є і завдання синтезу булевих функцій, які можуть

служити в якості функціональної основи нових, стійких до розкриття криптографічних алгоритмів. Булеві функції, що володіють специфічними криптографічними властивостями є функціональною основою генераторів псевдовипадкових послідовностей, орієнтованих на використання в системах захисту інформації.

Дослідження криптографічних властивостей булевих функцій активно почалося з другої половини 80-х років - основні зусилля дослідників були сконцентровані на з'ясування принципів, покладених в основу конструювання S-блоків. До середини 90-х років дослідження криптографічних властивостей булевих функцій практично розкрило закони побудови S-блоків. Великі успіхи зроблені в напрямку побудови булевих функцій для генераторів псевдовипадкових послідовностей. Основна частина публікацій присвячена синтезу булевих функцій з заданими криптографічними властивостями. Однак більша частина цих досліджень носить теоретичних характер і отримані результати не враховують технологічні можливості їх ефективної реалізації універсальними і спеціалізованими засобами обчислювальної техніки. Проблема полягає в тому, що в даний час і в найближчому майбутньому потрібно синтезувати функції від великого числа змінних (більше 100) і велика частина розроблених методик в тій чи іншій формі припускають зберігання в пам'яті таблиць істинності, тобто вимагають пам'яті, об'єм якої якої екпоненціально залежить від числа змінних, що не може бути реалізовано на практиці. З цієї ж причини не вирішені в практичному плані і більшість завдань аналізу - тобто визначення криптографічних властивостей у заданих функцій [18]. Крім того, багато із запропонованих методів синтезують булеві функції, які не можуть бути ефективно реалізовані на універсальному процесорі, архітектура якого погано пристосована для обчислення значень булевих функцій.

Перш ніж безпосередньо приступити до вирішення шляхом розв'язку зазначених невирішених проблем, видається доцільним більш докладно розглянути сутність основних криптографічних властивостей булевих функцій.

Виходячи з теоретико-інформаційного підходу до процесів криптографічного кодування, суттєвим є те, що для забезпечення стійкості до спроб розкриття, всі використовувані в криптографічних алгоритмах булеві функції повинні володіти максимумом повної ентропії або, що те ж саме, бути балансними (тобто приймати одиничні значення на рівно половині з 2^n можливих наборів вхідних змінних) з тим, щоб їх значення-розряди вихідного коду були максимально навантаженими в інформаційному плані. Формально, булева функція $f(x_1,..., x_n)$ є балансною, якщо виконується умова:

$$\sum_{x_1,\dots,x_n \in Z} f(x_1, \dots, x_n) = 2^{n-1}, \qquad (1.1)$$

де Z - множина всіх 2^n можливих наборів значень n булевих змінних $x_1,...,x_n$. Використання балансних булевих функцій робить неефективним використання статистичних методів аналізу криптографічних алгоритмів.

Іншою важливою криптографічною властивістю булевих функцій, що роблить неефективним виключення змінних є умова максимуму умовної ентропії: $H(f(x_1, ... , x_q, ... , x_n)/ f(x_1, ..., x_q \oplus 1, ... , x_n))$, сенс якої полягає в тому, що будь-яка функція, що виходить при виключенні будь-якої зі змінних також повинна повинна задовольняти умові максимуму ентропії, тобто бути балансною.

Формальне визначення булевої функції, що задовольняє SAC може бути виконано наступним чином. Булева функція f(x) визначена на множині Z 2^n наборів n вхідних змінних $x_1, x_2, ..., x_n, x_j \in \{0,1\}, j = 1, ..., n$ задовольняє SAC-критерієм, якщо для кожної вхідної змінної змінної виконується умова:

$$\sum_{x \in Z} \Big(f\big(x_1, \dots, x_j, \dots, x_n\big) \oplus f\big(x_1, \dots, x_j \oplus 1, \dots, x_n\big) \Big) = 2^{n-1}, \qquad (1.2)$$

Для отримання систем некорельованих булевих функцій часто використовують булеві SAC-функції високих порядків.

Булева функція f(x) задовольняє SAC-критерієм m-го порядку, тоді і тільки тоді, коли виконуються наступні умови:

- Булева функція f (x) задовольняє критерію строгого лавинного ефекту для всіх змінних на яких вона визначена;
- Будь-яка булева функція, що отримується з f (x) шляхом фіксації будь-яких m вхідних змінних задовольняє SAC.

З практичної точки зору важливим завданням є отримання не просто SAC-функції, а SAC-функції з якомога більшою мірою нелінійності.

Відомо, що будь-яка булева функція f (x1, ..., xn) може бути представлена в алгебраїчній нормальній формі (алгебри Жегалкіна) у вигляді:

$$f(x_1, \dots, x_n) = a_0 \oplus a_i \cdot x_i \oplus a_{ij} \cdot x_i \cdot x_j \oplus \dots \oplus a_{1,2,\dots,n} \cdot x_1 \cdot x_2, \dots, \cdot x_n, \qquad (1.3)$$
$$i = 1 \dots n \quad 1 \le i < j \le n$$

Лінійної називають функцію, яка не містить добутків змінних в алгебраїчній нормальній формі (1.3). Ступенем нелінійності d(f(x)) функції f(x) називається максимальне число змінних в ненульовому добутку алгебраїчної нормальної форми представлення функції f(x) [19]. Нелінійністю N(f(x)) булевої функції f(x) називається Хеммінгова відстань вектора значень функції до вектора значень найближчій лінійної функції, тобто N(f(x)) визначається виразом:

$$N\Big(f\big(\underline{x}\big)\Big) = min \sum_{a_1, a_2, \dots, a_n \; x \in Z} \Big(f\big(\underline{x}\big) \oplus (a_0 \oplus a_1 \cdot x_1 \oplus \dots \oplus a_n \cdot x_n) \Big), \qquad (1.4)$$

Отримання SAC-функції з можливо великими значеннями d(f(x)) і N(f(x)) забезпечує стійкість функції перетворення до розкриття методами лінійної апроксимації.В роботі показано, що на множині всіх можливих булевих функцій від n змінних завжди можна вказати клас булевих функцій, що володіють максимальною нелінійністю, які отримали назву bent-функції [20]. Доведено, що максимальне значення нелінійності $N(f(\underline{x}))_{max}$ булевих функцій від заданого числа змінних, яке досягається тільки для bent-функцій визначається наступним виразом:

$$N(f(\underline{x}))_{max} = 2^{n-1} - 2^{n/2-1}, \quad (1.5)$$

Отримання булевих функцій, що задовольняють наведеним вище критеріям є досить складною і актуальною проблемою розвитку систем захисту інформації в базах даних, рішення якої дозволить підвищити ефективність алгоритмів криптографічного захисту інформації без додаткового ускладнення структури обчислень, а значить, без втрати в швидкості реалізації криптографічних алгоритмів.

1.2. Властивості криптографічних булевих функцій що володіють максимумом повної і умовної ентропії.

Поняття критерію строго лавинного ефекту (Strict Avalanche Criterion - SAC) було введено в криптографію Вебстером А.Ф. і Таверсом С.Є.

Криптографічне значення SAC визначається необхідністю для багатьох завдань криптографії відображення n біт вхідної інформації в одному єдиному біті. Істотним тут є те, щоб "відображення" носило можливо складніший характер. Тут поняття "складність" стосовно аналізованої завдачі не визначається строго математично.

Поняття критерію строго лавинного ефекту (Strict Avalanche Criterion - SAC) було введено в криптографію Вебстером А.Ф. і Таверсом С.Є.

Криптографічне значення SAC визначається необхідністю для багатьох завдань криптографії відображення n біт вхідної інформації в одному єдиному біті. Істотним тут є те, щоб "відображення" носило можливо складніший характер. Тут поняття "складність" стосовно аналізованої завдачі не визначається строго математично [21].

Інтуїтивне представлення складності пов'язує складність з відомою з теорії інформації вимогою отримання максимальної ентропії: H (f (x)), яка досягається, якщо сама функція f (x) є балансною. Однак, максимум значення ентропії не є достатньою умовою, щоб функція вважалася складною. Максимум умовної ентропії: $H[f(x_1,x_2,...,x_i,...,x_n) / f(x_1, x_2,...,x_i,...,x_n)] = 2^{n-1}$, що виконується для всіх і ($1 \le i \le n$), досягається в SAC-функції.

Зафіксувавши одну з вхідних змінних рівною 0 і 1, можна отримати з SAC-функції, визначеної на n змінних, дві підфункції f' і f'', визначені на (n-1)-й змінній, причому кожна з підфункцій буде балансною і містить інформацію, відмінну від тієї, яка міститься в іншій. Будь-яка підфункция f'завжди буде відносно поганим наближенням функції f за умови, що остання є SAC.

Дійсно, значення f 'відмінні від значень f з ймовірністю 1/4. Цей недолік точності апроксимуючих функцій меншої розмірності є властивістю, яка широко використовується в криптосистемах: існування деяких (відносно точних) апроксимуючих функцій меншої розмірності, ніж функція криптографічного перетворення, може зменшити обсяг роботи при підборі повнорозмірної функції криптографічного перетворення [22].

Функції, для яких перемикання кожного вхідного біта викликає перемикання вихідного біта, значно складніше апроксимувати (найкращі підфункції меншої розмірності збігаються по виходу з материнською функцією тільки в 50% випадків), проте, їх умовна ентропія дорівнює нулю.

Криптографічні властивості булевих функцій, що володіють максимумом умовної ентропії можуть бути досліджені за допомогою розгляду їх Уолш-спектра. Для отримання спектральної характеристики булевої функції f(x) необхідно виконати пряме перетворення Уолша відповідно до виразу:

$$F(w) = \sum_{x \in V_2^n} f(x) \cdot (-1)^{x*w}, \quad (1.6)$$

де x · w - позначає добуток x на w:

$$x * w = x_1 \cdot w_1 + x_2 \cdot w_2 + \cdots + x_n \cdot w_n, \quad (1.7)$$

Функція f(x) може бути відтворена по її спектру F(w) шляхом використання зворотного перетворення Уолша:

$$f(x) = 2^{-n} \sum F(w) \cdot (-1)^{x*w}, \quad (1.8)$$

На практиці роботи зі спектрами простіше використовувати зміщені булеві функції, які приймають значення на множині {1, -1}. Зміщена булева функція f^(x) визначається у вигляді:

$$f^{\wedge}(x) = (-1)^{f(x)} \text{ або } f^{\wedge}(x) = 1 - 2 \cdot f(x), \quad (1.9)$$

Відношення між перетвореннями Уолша для зміщеної f ^ (x) і незміщеної булевої функції f (x) задаються наступними лемами:

Лемма 1: Якщо зміщена булева функція f ^ (x) = (-1) f (x), то зміщений спект Уолша визначається через спектр звичайної функції у вигляді

$$F^{\wedge}(w) = -2F(w) + 2^n \cdot \delta(w), \quad (1.10)$$

Відповідно, спектр F (w) звичайної (незміщенної) булевої функції може бути виражений через спектр Уолша зміщеної булевої функції наступним чином:

$$F(w) = 2^{n-1} \cdot \delta(w) - 1/2 \ F^{\wedge}(w), \quad (1.11)$$

$$\text{де } \delta(w) = \begin{cases} 1, \text{ для } w = 0 \\ 0, \text{ у всіх інших випадках} \end{cases}$$

Нехай x и x_i позначають два n-розрядних вектора, що відрізняються тільки в i-м розряді.

f(x) задовольняє SAC, якщо виконується наступна умова:

$$\sum_{x \in Z_2^n} f(x) \oplus f(x_j) = 2^{n-1}, \text{ для будь} - \text{якого } i: 1 \le i \le n, \quad (1.12)$$

Якщо позначити через c_i n-розрядний двійковий вектор з 1-цею у і-му розряді і нулями у всіх інших, то умову SAC можна записати у вигляді:

$$\sum_{x \in Z_2^n} f(x) \oplus f(x \oplus c_i) = 2^{n-1}, \text{ для всіх } i: 1 \le i \le n, \quad (1.13)$$

Для зміщеної булевої функції f^(x) = (-1)f(x) умову (1.13) можна сформулювати у вигляді:

$$\sum_{x \in Z_2^n} f\hat{}(x) \cdot f(x \oplus c_i) = 0, \quad (1.14)$$

Це означає, що на рівно половині можливих значень вектора x логічний добуток f^(x) · f^(x ⊕ ci) = -1, а на решті половини можливих значень вектора x: f^(x) · f^(x ⊕ c_i)=1.
Ліва частина рівняння (1.14) може бути представлена з використанням згортки в наступному вигляді:

$$\sum_{x \in Z_2^n} f\hat{}(x) \cdot f\hat{}(x \oplus c_i) = [f\hat{} * f\hat{}] \cdot (c_i), \quad (1.15)$$

де через символ * позначено векторний добуток, а через символ · позначено скалярний добуток.
За відомою теоремою про властивості згортки отримаємо:

$$h(x) = \sum_{y \in Z_2^n} f(y) \cdot g(y + x) \Leftrightarrow H(w) = F(w) \cdot G(w), \quad (1.16)$$

Отже, теорема про згортку говорить, що якщо булева функція h (x) виходить у вигляді:

$$h(x) = \sum_{y \in Z_2^n} f(y) \cdot g(y + x), \quad (1.17)$$

то Уолш-перетворення функцій h (x), f (x) і g (x) пов'язані між собою таким співвідношенням:

$$H(w) = F(w) \cdot G(w), \tag{1.18}$$

З цієї теореми видно, що ліва частина рівняння (1.18) може бути виражена через інверсне Уолш-перетворення:

$$F\hat{}(w) \cdot F\hat{}(w) = F\hat{}^2(w), \tag{1.19}$$

Використовуючи формулу зворотного Уолш- перетворення:

$$f(x) = 2^{-n} \sum_{w \in Z_2^n} F(w) \cdot (-1)^{x \cdot w}, \tag{1.20}$$

можна отримати такий вираз:

$$\begin{aligned} &[f\hat{} * f\hat{}] \cdot (c_i) \\ &\quad = 2^{-n} \sum_{w \in Z_2^n} F\hat{}^2(w) \cdot (-1)^{c(i) \cdot 2} \\ &\quad = 2^{-n} \sum_{w \in Z_2^n} F\hat{}^2(w) \cdot (-1)^{w(i)}, \end{aligned} \tag{1.21}$$

Використовуючи F(w) = -2F(w) + $2^n \cdot \delta(w)$,, можна отримати наступну теорему, яка може служити основою для синтезу SAC-функцій :

Теорема 1: Зміщенна булева функція f^(x), визначена на просторі $z_2^n \rightarrow \{1 , -1\}$, відповідає критерію максимуму умовної ентропії, тобто SAC тоді і тільки тоді, коли її Уолш-представлення F^(w) задовольняє умові :

$$\sum_{w \in Z_2^n} (-1)^{w(i)} \cdot F\hat{}^2(w) = 0, \tag{1.22}$$

для всіх значень $i \in \{1, 2, \ldots, n\}$

Застосовуючи до прямого Уолш-перетворення F(w) незміщенної булевої функції f(x) = 1/2 · (1 - f^(x)) умова відповідності критерію максимуму умовної ентропії або SAC має вигляд:

$$\sum_{w \in Z_2^n} (-1)^{w(i)} \cdot F^2(w) = 2^n \cdot F([0, \ldots, 0]) - 2^{2n-2}, \tag{1.23}$$

для усіх i ∈ {1, 2, . . . , n}

Важливою властивістю для визначення критерію максимуму повної ентропії через дослідження Уолш-представлень є те, що чисельне значення F ([0,..., 0]) відповідає числу одиниць у вихідній таблиці істинності булевої функції f (x) [23]. Приклад подання булевої функції f (x), що задовольняє критерію максимуму повної та умовної ентропії наведено в таблиці 1.1.

Таблиця 1.1.

Булева функція f (x), що задовольняє критерію максимуму повної та умовної ентропії.

x1	x2	x3	Функція	w1	w2	w3	Уолш-спектр
			f(x)				F(w)
0	0	0	0	0	0	0	4
0	0	1	1	0	0	1	0
0	1	0	1	0	1	0	-2
0	1	1	0	0	1	1	-2
1	0	0	0	1	0	0	0
1	0	1	0	1	0	1	0
1	1	0	1	1	1	0	2
1	1	1	1	1	1	1	-2

Геометрична інтерпретація наведеної теореми про властивості Уолш-представлення булевих функцій, що володіють максимумом умовної ентропії (SAC-функцій) може бути отримана, якщо розглядати n-мірний простір в координатах w1, w2, ..., wn .. а значення Уолш-спектра розглядати як деяку

"вагу", розташовану по кутах n-мірного куба. Тоді, якщо булева функція відповідає SAC-критерієм, то "центр ваги" фігури, побудованої по Уолш-спектру цієї функції знаходиться рівно в геометричному центрі n-мірного куба [24].

Розглянемо поняття 50% -вої залежності булевих функцій по відношенню до їх вхідних змінних. Саме ця концепція використовується для визначення SAC. Булева зміщена функція $f\hat{}(x) : z_2^n \to \{1,-1\}$ (і відповідно, звичайна булева функція $f(x) : z_2^n \to \{0,1\}$) називається на 50% залежною від і-тої вхідної змінної, якщо функція з ймовірністю 1/2 приймає значення на наборах, що відрізняються в і-тому розряді, тобто виконується умова:

$$\sum_{x \in Z_2^n} f\hat{}(x) \cdot f\hat{}(x \oplus c_i) = 0 \text{ або } \sum_{x \in Z_2^n} f(x) \oplus f(x \oplus c_i) = 2^{n-1}, \quad (1.24)$$

Булева функція відповідає критерію строго лавинного ефекту, або SAC тоді і тільки тоді, якщо вона має властивість 50% -вої залежності від кожної зі змінних.Наступна теорема визначає достатні умови для того, щоб функція була 50% - залежною від однієї або більше вхідних змінних.

Теорема 2: Якщо для вектора, не рівного нулю $c \in z_2^n$ і для всіх $w \in z_2^n$ виконується умова:

$$F\hat{}^2(w) = F\hat{}^2(w \oplus c), \quad (1.25)$$

і якщо вектор c містить m одиниць ($c_{i1} = c_{i2} = ... = c_{im} = 1$, $1 \le m \le n$), тоді $f\hat{}(x)$ є 50% -залежною від вхідних змінних $x_{i1}, x_{i2}, . . ., x_{im}$.

ВИСНОВКИ ДО РОЗДІЛУ 1

В результаті виконання досліджень, спрямованих на аналіз сучасного стану використання булевих функціональних перетворень в алгоритмічних засобах криптографічного захисту інформації, а також методів синтезу булевих функцій для таких перетворень можна зробити наступні висновки:

1. Виконано огляд методів криптографічних алгоритмів, що застосовуються в сучасних системах захисту інформації в комп'ютерних системах і мережах, проведений оглядовий аналіз використання методів функціонального аналізу для визначення стійкості до розкриття широкого класу криптографічних алгоритмів. В результаті аналізу показано, що стійкість широкого класу криптографічних алгоритмів, в основі яких важковирішувана математична задача відшукання коренів систем нелінійних булевих функцій визначається відповідністю булевих функцій бітового перетворення певним критеріям, одним з яких є критерій максимуму умовної ентропії або строгого лавинного ефекту. Показано, що задача синтезу таких функцій для створення криптостійкі алгоритмів захисту інформації є далеко не тривіальним завданням.
2. Виконано оглядовий аналіз існуючих методів отримання булевих функцій зазначеного класу, який дозволив виявити їх основні недоліки: велику трудомісткість і непридатність для синтезу функцій від великого числа змінних, і таким чином обгрунтована необхідність продовження робіт в цьому напрямку.

РОЗДІЛ 2
РОЗРОБКА МЕТОДУ ОТРИМАННЯ БУЛЕВИХ ФУНКЦІЙ, ЯКІ МАЮТЬ ЛАВИННИЙ ЕФЕКТ

2.1. Отримання небалансних SAC-функцій

При використанні цього принципу виключається поява варіанту підсумовування 2-х одиниць при підсумовуванні $f(x_1,...,x_i,...,x_n) \oplus f(x_1,...,1 \oplus x_i,...,x_n)$, для всіх $\{x_1,...,x_n\} \in Z$ і всіх значень $i \in \{1,..., n\}$, так що виключаються варіанти при яких як $f(x_1,...,x_i,...,x_n)=1$ і $f(x_1,...,1 \oplus x_i,...,x_n) = 1$.. Виходячи з цього принципу при підсумовуванні згідно (1.3) не можуть зникати одиниці і для того, щоб функція задовольняла б критерію максимуму умовної ентропії необхідно, щоб число наборів, на яких функція приймає одиничне значення в точності дорівнювала 2^{n-2}. У практичному плані, сутність методів отримання SAC, в основі яких лежить розглянутий принцип полягає в тому, що число наборів, на яких генерується булева функція $f(x_1,...,x_n)$ приймає одиничне значення в точності дорівнює 2^{n-2} і зазначені набори підбираються таким чином, щоб віиконувалась наступна умова : $\exists \{ x_1,...,x_i,...,x_n\}$: $f(x_1,...,x_i,...,x_n) = 1, \Rightarrow f(x_1,...,1 \oplus x_i,...,x_n) = 0$, $i = 1,..., n$, і для інших наборів $f(x_1,...,x_i,...,x_n)=0$ и $f(x_1,..., 1 \oplus x_i,...,x_n) = 0$. З цієї умови прямо випливає, що при підсумовуванні по всім 2^{n-1} можливим наборам булевих змінних $x_1,...,x_{i-1},x_{i+1},...,x_n$: $\sum f(x_1,...,x_i,...,x_n) \oplus f(x_1,...,1 \oplus x_i,...,x_n) = 1, \exists i \in \{1,...,n\}$, тобто така функція буде відповідади SAC-критерію. Очевидно, що розглянутий принцип суттєво спрощуючи синтез функцій, що відповідають SAC-критерієм, разом з тим не дозволяє генерувати всі можливі SAC-функції.

На основі наведеного принципу розроблений метод генерації SAC-функцій в значній мірі вільний від недоліків відомих методів.

Пропонований метод отримання SAC-функцій заснований на використанні таблиці істинності генерованої булевої функції, яка зберігається в масиві $F(x_1,...,x_n)$ і складається в послідовному виконанні наступних дій:

1. Значення генерованої булевої функції $F(x_1,...,x_n), x_1,...,x_n \in Z$, де Z - множина 2^n всіляких наборів n двійкових змінних $x_1,...,x_n$ вважаються невизначеними.
2. Довільним чином вибирається набір $x_1',...,x_n' \in Z$ на якому значення генерованої функції не визначено. Значення функції на зазначеному наборі встановлюється рівним 1, тобто приймається $F(x_1',...,x_n')=1$.
3. Для всіх наборів $x_1',...,x_i' \oplus 1,...,x_n'$, i = 1,...,n встановити $F(x_1',...,x_i' \oplus 1,...,x_n') = 0$. Пункти 2 і 3 повторити 2^{n-1} разів.
4. На всіх наборах булевих змінних $x_1,...,x_n$ для яких значення функції $F(x_1,...,x_n)$ залишилось невизначеним встановити $F(x_1,...,x_n)=0$.

Доведемо, отримана описаним способом функція задовольняє SAC-критерію. За визначенням, булева функція відповідає SAC-критерію, якщо виконується умова (1.3). Всі можливі набори булевих змінних $x_1,...,x_n$ Z розподілимо по трьом непересічним множинам: $\{x_1,...,x_n\}$: $F(x_1,...,x_n) = 1$; $\{x_1,...,x_n\}$: $F(x_1,...,x_n) = 0$, x_j, j {1,...n}: $F(x_1,...,x_j\ 1,...,x_n)=1$; $\{x_1,...,x_n\}$: $F(x_1,...,x_n) = 0$, x_j, j {1,...n}: $F(x_1,...,x_j\ 1,...,x_n) = 0$. Очевидно, що при такій розбивці множин всіляких наборів вхідних змінних = i Z. Розглянемо далі всі можливі варіанти суми по модулю 2: $F(x_1,...,x_j,...,x_n)$ $F(x_1,...,x_j\ 1,...,x_n)$, j=1,...,n. Таких варіантів існує три: при першому варіанті $\{x_1,..., x_j,...,x_n\}$ і $\{x_1,...,x_j\ 1,...,x_n\}$ - при цьому $F(x_1,...,x_j,...,x_n) = 1$ і $F(x_1,...,x_j\ 1,...,x_n) = 0$, (із $F\{x_1,...,x_j,...,x_n\}$, випливає, що обов'язково $\{x_1,...,x_j\ 1,...,x_n\}$ оскільки пунктом 3 наведеної вище методики формована функція приймається рівною нулю на всіх наборах, що відрізняються значенням однієї змінної від набору на якому ця ж функція приймає значення рівне одиниці). У другому варіанті має місце зворотна

ситуація: $\{x_1,...,x_j,...,x_n\}$ Θ і $\{x_1,...,x_j\ 1,...,x_n\}$. Очевидно, що в розглянутих варіантах $F(x_1,...,x_j,...,x_n)$ $F(x_1,...,x_j\ 1,...,x_n)=1$. Третій варіант передбачає ситуацію, при якій набір $\{x_1,...,x_j,...,x_n\}$, а набір булевих змінних, на якому визначено другий доданок: $F(x_1,...,x_j\ 1,...,x_n)$ при цьому може належати множині Θ або Ω, причому в обох випадках $F(x_1,...,x_j\ 1,...,x_n) = 0$ як і перший доданок, так, що в третьому варіанті $F(x_1,...,x_j,...,x_n)$ $F(x_1,...,x_j\ 1,...,x_n)=0$. При обчисленні суми $F(x_1,...,x_j,...,x_n)$ $F(x_1,...,x_j\ 1,...,x_n)$ на всіх можливих наборах $x_1,...,x_j,...,x_n$ загальне число доданків за першим або другим варіантами буде в точності дорівнює числу одиниць в таблиці істинності генерованої булевої функції, тобто 2^{n-2}, а на наборах по третьому варіанту значення доданків дорівнюватимуть нулю, так, що загальне значення суми дорівнюватиме 2^{n-2} , для всіх j = 1,..., n, а це відповідно до (1.2) означає, що згенерована функція відповідає критерію максимуму умовної ентропії (SAC-критерію).

Покажемо, що описаний метод конструктивний, тобто при його використанні не виникає тупикових ситуацій. В принципі джерелом можуть тільки дві конфліктні ситуації: при спробі проставити значення функції у нуль, на зазначеному наборі раніше значення функції встановлено в одиницю, а також ситуація, що виникає коли при спробі встановити значення функції в одиницю на одному з невизначених раніше наборів, множина невизначених наборів виявиться порожньою. Покажемо неможливість появи зазначених ситуацій. Перша конфліктна ситуація виникає в тому випадку, коли на незайнятому наборі булевих змінних $x_1',...,x_n'$ значення функції визначається рівним одиниці: $F(x_1',...,x_n')=1$ і тоді необхідно встановити на n наборах $\{x_1',...,x_i\ 1,...,x_n\}$, значення функції дорівнює 0. Конфлікт виникає, якщо k $\{1,...,n\}$: $F(x_1',...,x_k\ 1,...,x_n) = 1$. Однак, якщо раніше функція на наборі $\{x_1',...,x_k\ 1,...,x_n\}$ була прийнята рівною 1, то згідно п.3 на всіх наборах, що відрізняється значенням тільки однієї змінної функція, була раніше прийнята рівною одиниці і, отже набір $\{x_1',...,x_k,...,x_n'\}$ не міг залишитися незайнятим, а, отже, умови для виникнення першої конфліктної ситуації ніколи не можуть бути

виконані. Доказ неможливості другої з перерахованих вище ситуації істотно складніше. Для цього слід більш детально розглянути комбінаторне питання про кількість можливих наборів, віддалених один від одного на заданій хеммінговій відстані. Сутність запропонованого методу отримання SAC-функцій з числом одиниць, рівним 2^{n-2} фактично полягає у виборі множини наборів $\{x_1,...,x_n\}$: $\{x_1,...,x_n\}$, $F(x_1,...,x_n) = 1$, при чому $\underline{X} = \{x_1,...,x_n\}$ і $\underline{Y} = \{y_1,...,y_n)$ виконується умова:

$$\sum_{i=1,...,n} (x_i \oplus y_i) \geq 2 \qquad (2.1)$$

Фактично сформована множина ϑ є підмножиною множини Ξ наборів віддалених один від одного на хемміноговій відстані, що перевищує 1, тобто $\vartheta \in \Xi$.

Формально, множину Ξ можна визначити наступним чином: $\underline{X}=\{x_1,...,x_n\}$ і $\underline{Y} = \{y_1,...,y_n)$ виконується умова (2.1), причому не існує такого набору $\{y_1,...,y_n\}$ Z, що не належить Ξ для якого виконувалося б умова (2.1). Слід зазначити, що множина Ξ є багатоваріантною, причому кожен з варіантів зазначеної множини відрізняється не тільки наборами булевих змінних, що в нього входять, а й різною їх кількістю. Таким чином, можна говорити про деяку множину Ψ, що включає всі w можливі множини Ξ_j, j = 1,...,w наборів, віддалених один від одного на хеммінговій відстані більшій 1. Очевидно, досліджуваний метод отримання SAC-функцій є конструктивним, тобто не породжує тупикових ситуацій, якщо кількість наборів у всіх множинах $_j$ не менш 2^{n-2}. Для j-ої множини $_j$ кількість наборів $\{x_1,...,x_n\}$ $_j$ і число одиничних компонент яких дорівнює k позначимо через $\xi_j(k, n)$. Очевидно, що загальне число S_j елементів множини Ξ_j дорівнює $S_j = \xi_j(0,n) + \xi_j(1,n) +....+ \xi_j(n,n)$. Максимальне значення $\xi_j(k,n)$ визначається числом можливих варіантів n-розрядних двійкових наборів, що містять рівно k одиниць і n-k нулів, тобто C^k_n. Всі можливі множини $\Xi \in \Psi$ можна умовно розділити на дві групи: до

першої слід віднести такі множини Ξ_j для яких чисельні значення $\xi_j(k,n)$ {0, C^k_n}, при цьому якщо $\xi_j(k,n)= C^k_n$,то $\xi_j(k+1,n)=0$ і $\xi_j(k-1,n)=0$. Якщо $\xi_j(0,n)=1$, то для всіх парних k $\xi_j(k,n)= C^k_n$, а для всіх непарних k: $\xi_j(k,n)=0$, і якщо $\xi_j(0,n)=0$, то для вісх парних k $\xi_j(k,n)=0$, а для всіх непарних k : $\xi_j(k,n)$ C^k_n. Відповідно, сумарне число наборів у множинах Ξ першої групи в обох випадках дорівнює 2n-1. Таким чином, в цих випадках $S_j= 2^{n-1}$. $\xi_j(k,n)$ 2^{n-2} і, отже, тупикових ситуацій неможливості вибору множини ϑ не може виникнути. Більш складно оцінити сумарне число наборів в множині Ξ в другому випадку, тобто коли $\xi_j(k,n) < C^k_n$. Без порушення спільності будемо вважати, що множина Ξ_j визначається послідовно, починаючи з k = 0 до k = n. Тоді можна вважати, що значення $\xi_j(k, n)$ залежить від $\xi_j(k-1, n)$, причому зазначена залежність не є однозначною, оскільки велику роль відіграє вибір $\xi_j(k-1, n)$ наборів. Тому при оцінці зазначеної залежності слід окремо розглядати функцію залежності верхньої границі значень $\sup(\xi_j(k,n)) = fs(\xi_j(k-1,n))$ і функцію залежності нижньої границі - $\inf(\xi_j(k,n))= f_i(\xi_j(k-1,n))$. Функція $\sup(\xi_j(k,n))$ реалізується в разі вибору $\xi_j(k-1,n)$ наборів з мінімальною їх сумарною хеммінговою відстанню, а функція $\inf(\xi_j(k,n))$ досягається при виборі $\xi_j(k-1,n)$ наборів з максимальною їх сумарною хеммінговою відстанню. Функції $\sup(\xi_j(k,n)) = fs(\xi_j(k-1,n))$ і $\inf(\xi_j(k,n)) = fi(\xi_j(k-1,n))$ можуть бути отримані аналітичним шляхом аналізу варіантів вибору k-1 одиниць в n-розрядному двійковому коді і визначення числа кодів, що забороняються (тобто відрізняються тільки в одному розряді), що містять k одиниць в n-розрядному двійковому слові. Однак аналітичне визначення розглянутих функцій досить громіздке через багатоваріантності вибору наборів, що містять k-1 одиницю і тому не може бути виконано в достатній мірі коректно для булевих функцій з великим числом змінних. З практичної точки зору важливим є визначення граничних чисел $k_{гр}$ (для заданого n) при яких аналізовані функції звертаються в нуль, тобто $fs(\xi_j(k_{грs}-1,n)) = 0$ і $\inf(\xi_j(k_{грi}-1,n)) = 0$. Нескладно показати, що функція

fs(ξ_j(k-1,n)) = 0 при k >= $k_{грs}$ = C_n^{k-1} -3, наприклад при n = 10 число можливих наборів з 3-ма одиницями перетворюється в нуль при включенні до складу множини Ξ більш $k_{грs}$ = 43 наборів з двома одиницями. Більш складно визначити залежність від k перетворення в нуль функції inf($_j$(k,n)). Можна показати, що значення $k_{грі}$ може бути визначене в такий спосіб:

$$k_{\text{грі}} = \frac{C_n^{k-1}}{\delta(n)}, \delta(n) = \sum_{j=1,\dots,C(k-1,n)} S_j(n), \qquad (2.2)$$

де S_j(n) - кількість n-розрядних двійкових кодів, що містять k одиниць, які знаходяться на хеммінговій відстані, що дорівнює 1 від j-го n-розрядного коду, що містить рівно k-1 одиницю і не знаходяться на такій хеммінговій відстані від усіх j-1 раніше обраних наборів з k-1 одиницею за умови, що сумарна хеммінгова відстань j-го коду з k-1 одиницею до всіх раніше обраних наборів максимальна. Очевидно, що S(1)= n-k; S(2)=n-k-1, якщо n/k 2, S(3) = n-k-1, якщо n/k 3, тобто можливий вибір такого 3-го набору, що містить k-1 одиниць, що він буде перебувати на хеммінговій відстані k-1 раніше обраних двох наборів.

При подальшому збільшенні j значення S (j) зменшується в силу того, що зменшується сумарна хеммінгова відстань від обраного на j-му кроці наборі до раніше обраних на попередніх кроках наборів. У границі, по мірі збільшення j величина S(j) прямує до 1. Так як зі збільшенням n кількість таких елементів зростає, то функція (n) з збільшенням n має тенденцію до зменшення. Можна показати, що верхня границя sup((n)) обмежена величиною 2, так, що мінімальне значення inf($k_{грі}$) дорівнює C_n^{k-1}/2. Відповідно, найменше число n-розрядних двійкових чисел, що відрізняються один від одного не менш, ніж у двох розрядах досягається коли для парних значень k число обираних наборів дорівнює inf(ξ_j($k_{грі}$,n)), а число вибираних наборів для непарних чисел k дорівнює відповідно нулю (або навпаки - для непарних k число вибираних наборів рівно inf(ξ_j($k_{грі}$,n)), а для парних - нулю).

У будь-якому випадку, мінімальне сумарне число Qmin вибираних в множину Ξ наборів вхідних змінних визначається наступним виразом:

$$
\begin{aligned}
Q_{min} &= \sum_{k=2,4,\dots,en} \inf\left(\xi_i\left(k_{\text{грі}}, n\right)\right) \\
&= \sum_{k=2,4,\dots,en} \frac{C_n^{k-1}}{2} \\
&= \sum_{k=1,3,\dots,cn} \inf\left(\xi_i\left(k_{\text{грі}}, n\right)\right) \\
&= \sum_{k=1,3,\dots,cn} \frac{C_n^{k-1}}{2} = 2^{n-2},
\end{aligned}
\tag{2.3}
$$

де en = n, якщо n-парне і дорівнює n-1, якщо n-непарне, cn = n якщо n-непарне і дорівнює n-1 в іншому випадку.

Таким чином показано, що при будь-якому порядку вибору n-розрядних двійкових кодів, що відрізняються один від одного не менш ніж в 2-х розрядах, сумарне число обраних кодів завжди буде знаходитися в інтервалі від 2^{n-2} до 2^{n-1}. Відповідно, при виборі 2^{n-2} наборів володіють зазначеною властивістю ніколи не може виникнути тупикової ситуації коли безліч можливих виборів виявиться порожнім. Звідси випливає, що пропонований метод отримання SAC-функцій є конструктивним.

Нижче наведено приклад генерації SAC-функції 5-ти змінних (n = 5) з використанням запропонованої методики синтезу таких булевих функцій. Процес побудови SAC-функції відображено в наведеній нижче таблиці 2.1.

Таблиця 2.1.

SAC-функція від 5-ти змінних.

Номер кроку	Вибраний набір, на якому F=1	Відмічені набори, на яких F=0
1	{0,1,0,0,1}	{0,1,0,0,1},{0,1,0,0,1},{0,1,1,0,1},{0,1,0,1,1}, {0,1,0,0,0}
2	{1,0,1,1,0}	{0,0,1,1,0},{1,0,1,0,0},{1,1,1,1,0},{1,0,0,1,0}, {1,0,1,1,1}
3	{1,0,1,0,1}	{0,0,1,0,1},{1,1,1,0,1},{1,0,0,0,1},{1,0,1,1,1}, {1,0,1,0,0}
4	{ 0,1,1,1,0 }	{1,1,1,1,0},{0,0,1,1,0},{0,1,1,0,0},{0,1,0,1,0},{0,1,1,1,1}
5	{ 0,0,0,1,1 }	{1,0,0,1,1},{0,1,0,1,1},{0,0,0,0,1},{0,0,0,1,0}, {0,0,1,1,1}
6	{ 1,1,1,1,1 }	{0,1,1,1,1},{1,0,1,1,1},{1,1,0,1,1},{1,1,1,0,1}, {1,1,1,1,0}
7	{ 0,0,0,0,0 }	{1,0,0,0,0},{0,1,0,0,0},{0,0,1,0,0},{0,0,0,1,0}, {0,0,0,0,1}
8	{ 1,1,0,0,0 }	{0,1,0,0,0},{1,0,0,0,0},{1,1,1,0,0},{1,1,0,1,0},{1,1,0,0,1}

Згенерована булева функція відповідає SAC-критерію і представлення її в алгебраїчній нормальній формі має такий вигляд:

$F=1 + x_1 + x_2 + x_3 + x_4 + x_5 + x_1x_3 + x_1x_4 + x_1x_5 + x_2x_3 + x_2x_4 + x_3x_4 + x_3x_5 + x_2x_4x_5 + x_1x_2x_5$

Нелінійність отриманої SAC-функції дорівнює 8-ми.

Визначимо найважливіші криптографічні характеристики булевих SAC-функцій, що генеруються пропонованим методом. Очевидно, що генеровані SAC-функції не є балансними в силу того, що число одиниць, які проставляються в таблиці істинності генерованої функції в точності дорівнює 2^{n-2}.

Іншою важливою в криптографічному відношенні характеристикою генерованих булевих SAC-функцій алгоритмів захисту даних є їх нелінійність. Так як генерована SAC-функція містить 2^{n-2} одиниць в таблиці істинності, а будь-яка лінійна функція - 2^{n-1}, то нелінійність отриманої з використанням запропонованого методу булевої SAC-функції буде не нижче 2^{n-2}. Однак, якщо оптимізувати вибір наборів генерованої функції на яких ця функція приймає одиничне значення, то нижня межа нелінійності може бути істотно підвищена і, відповідно підвищиться крипостійкість алгоритму проти розкриттів методами лінійного криптоаналізу. Для цього необхідно використовувати функцію E(t, k, n) для лінійних функцій. Функція E (t, k, n) - залежності числа наборів що містять рівно t одиниць на яких лінійні функції, АНФ яких містить рівно k термів дозволяє оцінити нижню межу нелінійності і оптимізувати вибір наборів, на яких генерована функція приймає одиничне значення за критерієм максимуму досягаємої нелінійності. Нехай генерована булева SAC-функція $f(x_1,...,x_n)$ характеризується введеною вище функцією $\xi(t,n)$, що визначає число наборів, з одиничними значеннями рівно t змінних на яких згенерована булева функція $f(x_1,...,x_n)=1$. Очевидно, що нижня межа нелінійності $N(f(x_1,...,x_n))$ цієї булевої функції може бути визначена в наступному вигляді:

$$N(f(x_1,\dots,x_n) \geq R\big(f(x_1,\dots,x_n)\big) = min \sum_{k=1,\dots,n\ t=0,\dots,n} |\xi(t,n) - E(t,k,n)|), \quad (2.4)$$

Таким чином, при виборі 2^{n-2} наборів на яких генерована функція приймає одиничне значення необхідно вибирати функцію $\xi(t, n)$ розподілу наборів по числу t змінних, що приймають середнє арифметичне значення, щоб значення $R (f (x_1,...,x_n))$ було б максимальним.

Використання такого підходу дозволяє підвищити рівень нелінійності генерованих булевих SAC-функцій. Так, вище було показано, що для булевих функцій від 6 змінних нижня межа нелінеості становить $2^4 = 16$, проте за рахунок спеціального вибору функцій $\xi(t, n)$ нижня межа нелінійності може бути помітно знижена. Наприклад, при використанні значень функції: $\xi(0,6)=0$, $\xi(1,6)=1$, $\xi(2,6)=7$, $\xi(3,6)=0$, $\xi(4,6)=7$, $\xi(5,6)=1$, $\xi(6,6)=0$ значення нижньої границі $R(f(x_1,...,x_6))$ нелінійності становить 20.

Відповідно до зазначеної функцією розподілу кількості одиниць по викладеній вище методиці можна завжди побудувати булеву функцію, що відповідає критерію строго лавинного ефекту (тобто SAC-функцію). Наприклад, наведеній вище булевої функції $\xi(t, 6)$ відповідає булева функція, яка приймає одиничне значення на наборах: { 0, 0, 0, 0, 0, 1 } для якого t=0, { 0, 0,1, 0, 1, 0}, { 0, 0, 1, 1, 0, 0 }, { 0, 1, 0, 0, 1, 0 }, { 0, 1, 0, 1, 0, 0 }, { 0, 1, 0, 1, 0, 0 }, { 0, 1,1,0,0,0}, {1,0,0,0,1,0} для яких t=2, {0,0,1,1,1,1}, {0,1,0,1,1,1}, {0,1,1,0,1,1}, {0,1,1,1,1,0}, {1,0,0,1,1,1}, {1,0,1,0,1,1} для яких t=5, {1,1,1,1,0,1} - для якого t=5. Алгебраїчна нормальна форма в алгебрі Жегалкіна цієї булевої функції, що задовольняє критерію строгої лавинності (SAC-критерію) має наступний вигляд: $F=x_6 + x_1x_6 + x_2x_6 + x_3x_6 + x_4x_6 + x_5x_6 + x_4x_5 + x_3x_5 + x_3x_4 + x_2x_5 + x_2x_4 + x_2x_3 + x_1x_5 + x_1x_2x_6 + x_1x_3x_6 + x_1x_4x_6 + x_1x_3x_4 + x_3x_4x_5 + x_1x_2x_4 + x_2x_4x_5 + x_1x_2x_3 + x_2x_3x_4 + x_2x_3x_5 + x_3x_4x_5x_6 + x_2x_4x_5x_6 + x_1x_2x_3x_4 + x_2x_3x_5x_6 + x_2x_3x_4x_5 + x_1x_2x_5x_6 + x_1x_3x_4x_5 + x_2x_3x_4x_5x_6 + x_1x_2x_4x_5x_6 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_4x_5 + x_1x_3x_4x_5x_6 + x_1x_2x_3x_4x_6$.

Нелінійність $N(F(x_1,x_2,...,x_6)$ згенерованої булевої функції $F(x_1,x_2,...,x_6)$, що відповідає критерію строгого лавинного ефекту за всіма 6-змінним дорівнює 20, при тому, що максимальне значення нелінійності для булевих функцій від 6-ти змінних дорівнює 26.

Викладена методика отримання SAC-функцій може бути модифікована з тим, щоб не використовувати пам'ять для збереження таблиці істинності

генерованої булевої SAC-функції, а зберігати тільки коди наборів, на яких синтезована булева функція приймає одиничне значення. Це дозволить зменшити необхідні обсяги пам'яті, що особливо актуально для побудови SAC-булевих функцій від великого числа змінних, що є характерним для сучасних алгоритмів криптографічного захисту інформації.

Перевагою запропонованої методики побудови SAC-функцій є простота і істотно менші витрати машинного часу в порівнянні з відомими методами вирішення цього завдання, що використовують властивість спектральної симетрії Уолш-представлень булевих функцій, що володіють максимумом умовної ентропії, оскільки реалізація зворотного Уолш-перетворення вимагає значних обчислювальних витрат, до того ж не завжди коректному Уолш-представленню відповідає дійсна булева функція (в цьому випадку виконання зворотного перетворення Уолша проводиться, але не породжує результуючої булевої SAC-функції).

До числа недоліків запропонованого методу отримання SAC-функцій слід, в першу чергу, віднести те, що генеровані булеві функції, відповідаючи критерієм максимуму умовної ентропії, не відповідають критерію максимуму повної ентропії, тобто не є балансними. Як буде показано нижче, такі функції можуть бути використані спеціальним чином для отримання балансних булевих функцій, що відповідають критерію строгої лавинності.

2.2. Розробка методики отримання балансних SAC-функцій

Важливою проблемою синтезу функцій перетворення інформації для криптографічних алгоритмів захисту даних є отримання булевих функцій, що задовольняють критерію максимуму повної та умовної ентропії тобто балансних SAC-функцій.

Основним підходом до синтезу таких функцій є використання різних рекурсій, коли балансна SAC-функція від n змінних виходить як результат певного функціонального перетворення $F(\phi(x_1,...,x_m),x_{m+1},...,x_m)$, где $\phi(x_1,...,x_m)$ - SAC-функція від меншого ніж n числа змінних.

Нижче пропонується метод отримання балансних SAC-функцій на основі небалансних SAC-функцій від n-2 змінних, які формуються за способом, викладеному в попередньому параграфі.

Без порушення загальності можна вважати, що для отримання булевої балансної SAC-функції $f(x_1,...,x_n)$ використовуються булеві функції f1, f2, f3, f4 від змінних x3,x4,...,xn і тоді булева функція $f(x_1,...,x_n)$ формується у вигляді:

$$f(x_1,\dots,x_n) = \overline{x_1}\cdot\overline{x_2}\cdot f1(x_3,\dots,x_n) \oplus \overline{x_1}\cdot x_2 \cdot f2(x_3,\dots,x_n) \oplus x_1\cdot\overline{x_2}\cdot f3(x_3,\dots,x_n) \oplus x_1\cdot x_2 \cdot f1(x_3,\dots,x_n) \quad (2.5)$$

Для формування булевих функцій $f1(x_3,...,x_n)$,..., $f4(x_3,...,x_n)$ використовуються дві допоміжні булеві функції $g(x_3,...,x_n)$ і $h(x_3,...,x_n)$, що визначаються наступним чином: $g(x_3,...,x_n)$ - булева функція n-2 змінних, балансна, але не SAC. Булева функція $h(x_3,...,x_n)$ відповідає SAC-критерію, приймає одиничне значення рівно на 2^{n-4} наборах, причому $g(x_3,...,x_n)\cdot h(x_3,...,x_n) = h(x_3,...,x_n)$ і $h(x_3\oplus 1,...,x_n\oplus 1)\cdot g(x_3,...,x_n) = 0$. Функції $f1(x_3,...,x_n)$,... ,$f4(x_3,...,x_n)$ формуються наступним чином:

$$f1(x_3,\dots,x_n) = g(x_3,\dots,x_n)\oplus h(x_3,\dots,x_n)\oplus 1$$
$$f2(x_3,\dots,x_n) = h(x_3\oplus 1, x_2\oplus 1\dots,x_n\oplus 1)$$
$$f3(x_3,\dots,x_n) = h(x_3,\dots,x_n) \quad (2.6)$$
$$f4(x_3,\dots,x_n) = g(x_3\oplus 1, x_2\oplus 1\dots,x_n\oplus 1)\oplus h(x_3\oplus 1, x_2\oplus 1\dots,x_n\oplus 1)\oplus 1$$

Покажемо, що булева функція f (x_1,...,x_n) побудована відповідно до виразів (2.5) і (2.6) є балансною і відповідає критерію SAC.

Булеві функції ($x_1 \oplus 1$)·($x_2 \oplus 1$)· f1(x_3,...,x_n) і x_1·($x_2 \oplus 1$)· f3(x_3,... ,x_n) приймають одиничні значення на 2^{n-4} наборах змінних x_1,...,x_n. Функції ($x_1 \oplus 1$)· x_2 · f2(x_3,...,x_n) і x_1· x_2· f4(x_3,...,x_n) приймають одиничні значення на 2^{n-2} - 2^{n-4} наборах. Оскільки дві або більше із згаданих функцій не можуть одночасно приймати одиничні значення на одному і тому ж наборі, то функція f (x_1,...,x_n) приймає одиничне значення на всіх наборах, на яких рівні одиниці зазначені вище функції і число таких наборів дорівнює 2^{n-1}, так що функція f (x_1,...,x_n) є балансною.

Покажемо, що функція f(x_1,...,x_n) відповідає критерію SAC по змінній x_1. Для цього розглянемо наступну функцію:

$$
\begin{aligned}
f(x_1 \oplus 1, x_2, \dots, x_n) \oplus f(x_1, \dots, x_n) \\
&= (x_1 \oplus 1) \cdot (x_2 \oplus 1) \\
&\cdot \big(f1(x_3, \dots, x_n) \oplus f3(x_3, \dots, x_n)\big) \oplus (x_1 \oplus 1) \cdot x_2 \\
&\cdot \big(f2(x_3, \dots, x_n) \oplus f4(x_3, \dots, x_n)\big) \oplus x_1 \cdot (x_2 \oplus 1) \\
&\cdot \big(f1(x_3, \dots, x_n) \oplus f3(x_3, \dots, x_n)\big) \oplus x_1 \cdot x_2 \\
&\cdot (f1(x_3, \dots, x_n) \oplus f3(x_3, \dots, x_n))
\end{aligned}
\qquad (2.7)
$$

У виразі (2.7) булеві функції ($x_1 \oplus 1$) · ($x_2 \oplus 1$) · (f1(x_3,...,x_n) ⊕ f3(x_3,...,x_n)) і x_1 · ($x_2 \oplus 1$) · (f1(x_3,...,x_n) ⊕ f3(x_3,...,x_n)) приймають середнє арифметичне значення на 2^{n-4} різних наборах, так як на такій кількості наборів змінних x_3,...,x_n приймає одиничне значення функція (f1(x_3,...,x_n) ⊕ f3(x_3,...,x_n)) = g(x_3,... ,x_n) ⊕ 1. Аналогічно, булеві функції, що входять у вираз (2.7) ($x_1 \oplus 1$) · x_2 · (f2(x_3,... ,x_n) ⊕ f4(x_3,... ,x_n)) і x_1 · x_2 · (f1(x_3,... ,x_n) ⊕ f3(x_3,...,x_n)) приймають одиничні значення на 2^{n-4} наборах. Отже функція, яка визначається виразом (2.7) приймає одиничні значення на 2^{n-1} наборах, а це означає виконання

умови (1.3), тобто функція $f(x_3,...,x_n)$ є SAC функцією по змінній x_1.

Для того, щоб показати, що допоміжна булева функція $f(x_3,...,x_n)$ отримана відповідно до (2.7) задовольняє SAC по змінній x2 слід розглянути функцію:

$$\begin{aligned} f(x_1, x_2 \oplus 1, \dots, x_n) \oplus f(x_1, \dots, x_n) \\ = (x_1 \oplus 1) \cdot (x_2 \oplus 1) \\ \cdot \big(f1(x_3, \dots, x_n) \oplus f2(x_3, \dots, x_n)\big) \oplus (x_1 \oplus 1) \cdot x_2 \\ \cdot \big(f1(x_3, \dots, x_n) \oplus f2(x_3, \dots, x_n)\big) \oplus x_1 \cdot (x_2 \oplus 1) \\ \cdot \big(f3(x_3, \dots, x_n) \oplus f4(x_3, \dots, x_n)\big) \oplus x_1 \cdot x_2 \\ \cdot (f2(x_3, \dots, x_n) \oplus f4(x_3, \dots, x_n)) \end{aligned} \tag{2.8}$$

Булеві функції $(x_1 \oplus 1)\cdot(x_2 \oplus 1) \cdot (f1(x_3,...,x_n) \oplus f2(x_3,... ,x_n))$ і $(x_1 \oplus 1) \cdot x_2 \cdot (f1(x_3,...,x_n) \oplus f2(x_3,...,x_n))$ приймають одиничне значення на 2^{n-4} різних наборах. Дійсно, функція $f1(x_3,... ,x_n) \oplus f2(x_3,...,x_n) = g(x_3,...x_n) \oplus h(x_3,...,x_n) \oplus h(x_3 \oplus 1,... ,x_n \oplus 1) \oplus 1$. Функція утворена сумою перших двох доданків приймає одиничне значення на 2^{n-2} наборах в силу того, що $g(x_3,...,x_n)\cdot h(x_3,...,x_n) = h(x_3,...,x_n)$.

Так як $h(x_3 \oplus 1,x_2 \oplus 1,...,x_n \oplus 1) \cdot g(x_3,...,x_n) = 0$ і функція $h(x_3 \oplus 1,x_2 \oplus 1,... , x_n \oplus 1)$ приймає одиничне значення на 2^{n-2} наборах, відмінних від тих, на яких приймає одиничне значення функція $g(x_3,...x_n) \oplus h(x_3,...,x_n)$ то булева функція $f1(x_3,... ,x_n) \oplus f2(x_3,... ,x_n)$ приймає одиничне значення рівно на 2^{n-1} наборі. Тоді булева функції $(x_1 \oplus 1) \cdot (x_2 \oplus 1) \cdot (f1(x_3,...,x_n) \oplus f2(x_3,... ,x_n))$ і $(x_1 \oplus 1) \cdot x_2 \cdot (f1(x_3,... ,x_n) \oplus f2(x_3,...,x_n))$ приймають середнє арифметичне значення на 2^{n-4} різних наборах. Аналогічним чином можна показати, що функція $f2(x_3,...,x_n) \oplus f4(x_3,...,x_n)$ також приймає одиничні значення на 2^{n-1} наборах і функції $x_1 \cdot (x_2 \oplus 1)\cdot(f3(x_3,... ,x_n) \oplus f4(x_3,... ,x_n))$ і $x_1 \cdot x_2 \cdot (f2(x_3,... ,x_n) \oplus f4(x_3,... ,x_n))$ приймають середнє арифметичне значення на 2^{n-4} різних наборах. Таким чином, функція

визначена (2.7) приймає одиничне значення на 2^{n-1} наборі, тобто являється балансною, а це значить, що булева функція $f(x_1,...,x_n)$ відповідає критерію SAC по змінній x_2.

Покажемо нарешті, що згенерована булева функція $f(x_1,...,x_n)$ відповідає критерію SAC по змінним $x_3,...,x_n$. Припустимо, що булеві функції $f1(x_3,...,x_n),..., f4(x_3,...,x_n)$ відповідають критеріям SAC по змінним $x_3,x_4,...,x_n$. Це означає, що, наприклад, для функції $f1(x_3,...,x_n)$ функція $(x_1 \oplus 1)\cdot (x_2 \oplus 1)\cdot f1(x_3,...,x_i,...,x_n) \oplus (x_1 \oplus 1)\cdot(x_2 \oplus 1) \cdot f1(x_3,...,x_i \oplus 1,...,x_n)$, $i \in \{3,...,n\}$ приймає одиничне значення на 2^{n-4} наборах. Аналогічний висновок справедливий і для інших булевих функцій $f2(x_3,...,x_n)$, $f3(x_3,x_4,...,x_n)$ и $f4(x_3,...,x_n)$, причому набори змінних на яких приймають одиничні значення булевих функції $(x_1 \oplus c_1)\cdot(x_2 \oplus c_2) \cdot (f_k(x_3,...,x_i,... ,x_n) \oplus f_k(x_3,...,x_i \oplus 1,... ,x_n))$, $c_1,c_2 \in \{ 0,1\}$, где $k=4 - 2\cdot c_1 + c_2$, $i \in \{3,...,n\}$ різні, так що загальне число наборів змінних на яких зазначені булеві функції приймають середнє арифметичне значення складе 2^{n-1}, так що результуюча булева функція отримана з використанням (2.5) $f(x_1, x_2,... x_n)$ відповідає критерію SAC по змінним $x_3,...,x_n$.

Доведемо тепер, що зроблене вище припущення про те, що булеві функції $f1 (x_3,...,x_n),..., f4 (x_3,...,x_n)$ відповідають критеріям SAC по змінним $x_3,...,x_n$. Так як функція $h (x_3,...,x_n)$ відповідає критерію SAC, то цим критерієм відповідає і симетрична їй функція $h (x_3 \oplus 1,...,x_n \oplus 1)$ - тому задовольняють критерію SAC функції $f3(x_1 ,...,x_n)$ і $f2(x_1,...,x_n)$, що збігаються відповідно з згаданими функціями. Функції $f1 (x_3,...,x_n)$ і $f4 (x_3,...,x_n)$ відповідають критерію SAC, якщо функція $r(x_3,... ,x_n) = h(x_3,... ,x_n) \oplus g(x_3,... ,x_n)$ відповідає критерію SAC. Ця умова еквівалентна умові: $g(x_3,... ,x_n) = h(x_3,... ,x_n) \oplus r(x_3,... ,x_n)$, де $r(x_3,...,x_n)$ - булева функція, що задовольняє SAC, приймаюча одиничні значення на 2^{n-2} наборах змінних $x1,...,x_n$, і така, що $h(x_3,...,x_n) \cdot r(x_3,...,x_n) = 0$.

Для того, щоб показати, що остання умова виконується, необхідно довести, що сума по модулю 2 булевих функцій $h(x_3,...,x_n) \oplus r(x_3,...,x_n)$ при зазначених вище умовах, що накладаються на $r(x_3,...,x_n)$ завжди буде балансною булевою функцією і не відповідати критерію строго лавинного ефекту, що сформульовано у вигляді такої теореми.

Теорема: Якщо $h(x_1,...,x_m)$ і $r(x_1,...,x_m)$ - дві булеві функції, визначені на множині значень m булевих змінних, що задовольняють SAC-критерію і приймають середнє арифметичне значення рівно на 2^{m-2} наборах, причому $h(x_1,...,x_m) \cdot r(x_1,...,x_m) = 0$, то функція $g(x_1,...,x_m) = h(x_1,...,x_m) \oplus r(x_1,...,x_m)$ є балансною і не задовольняє SAC-критерію за умови, що g (x1,...,xm) залежить від всіх m булевих змінних.

Доведення: Так як $h(x_1,...,x_m) \cdot r(x_1,...,x_m) = 0$ то $g(x_1,...,x_m) = h(x_1,...,x_m) \oplus r(x_1,...,x_m) = h(x_1,...,x_m) \vee r(x_1,...,x_m)$ і, оскільки кожна з функцій $h(x_1,...,x_m)$ и $r(x_1,...,x_m)$ приймає середнє арифметичне значення рівно на 2^{m-4} різних наборах, то функція $g(x_1,...,x_m)$ приймає одиничне значення рівно на 2^{m-1} наборах і, отже, є балансною.

Розглянемо множину ϑ_{h1} наборів m-1 змінних $x_1,...,x_m$ на яких булева функція $h(x_1,...,x_m)$ приймає одиничне значення: $\{x_2',...,x_m'\} \in \vartheta_{h1}$ якщо існує $x_1' \in \{0,1\}$ що $h(x_1',x_2'...,x_m') = 1$. Аналогічно визначимо множину ϑ_{r1} для $r(x_1,...,x_m)$: $\{x_2'',...,x_m''\} \in \vartheta_{r1}$: якщо існує $x_1'' \in \{0,1\}$ что $r(x_1'',x_2'',...,x_m'') = 1$. Очевидно, що в силу того, що функція $h(x_1,...,x_m)$ є SAC і приймає значення точно на 2^{m-2} наборах, то число наборів, що складають безмножинуліч ϑ_{h1} також дорівнює 2^{m-2} та всі набори цієї множини попарно різні. Аналогчний висновок справедливий і для множини ϑ_{r1}. Визначимо множину ϑ_{g1} наборів змінних $x_2,...,x_m$ таких, що $\{x_2''',...,x_m'''\} \in \vartheta_{g1}$ якщо існує $x_1''' \in \{0,1\}$ що $g(x_1''',...,x_m''') = 1$. Очевидно, що $\vartheta_{g1} = \vartheta_{h1} \cup \vartheta_{r1}$. Для того, щоб функція g(x1,...,xm) відповідала критерію SAC по змінної x_1 необхідно, щоб число

наборів, складових множини ϑ_{g1} дорівнювало 2^{m-2}. Ця умова може бути виконана тільки в тому випадку коли має місце $\vartheta_{h1} = \vartheta_{r1}$. У свою чергу, остання рівність можлива тільки за умови незалежності функції $g(x_1,...,x_m)$ від змінної x_1.

Таким чином, якщо булева функція $g(x_1,...,x_m)$ залежить від x_1, то вона не може відповідати критерію строго лавинного ефекту, тобто бути SAC-функцією. Поскільки вибір змінної x_1 в ході наведених вище теоретичних висновків був зроблений довільно, то аналогічні наведеним вище міркування будуть справедливі для будь-якої з m змінних, на яких визначені допоміжні булеві функції.

Наведена теорема дозволяє стверджувати, що булева функція $r(x_3,...,x_n) = h(x_3,...,x_n) \oplus g(x_3,...,x_n)$ відповідає критерію SAC і, отже, цьому критерію відповідають функції $f1(x_3,...,x_n)$ і $f4(x_3,...,x_n)$.

Таким чином, доведено, що булева функція $f(x_1,...,x_n)$ отримана відповідно до виразів (2.5) і (2.6) є балансною і відповідає критерію строгого лавинного ефекту, тобто є SAC (має максимальне значення умовної ентропії).

Практична реалізація наведеного методу зводиться до виконання наступних етапів:

- вибір функції $g(x_3,...,x_n)$;
- вибір функції $h(x_3,...,x_n)$;
- формування допоміжних булевих функцій $f1(x_3,...,x_n),..., f4(x_3,...,x_n)$ з використанням наведеного вище виразу (2.6) і отримання результуючої балансної булевої функції $f(x_1,.. .,x_n)$, що відповідає критерію строгої лавинності (тобто булевій функції, яка має максимальне значення повної та умовної ентропії по кожній із змінних) з виразу (2.5).

Нетривіальною є реалізація першого і другого етапів отримання SAC-функції.

Балансна функція g (x_3,...,x_n) що не задовольняє SAC і залежна від всіх змінних x_3,...,x_n може бути отримана наступним чином: довільно вибирається одна з змінних - наприклад x_3 і функція g(x_3,... ,x_n) формується у вигляді g(x_1,...,x_n) = u(x_4,...,x_n) $\oplus$ e(x_3,...,x_n) де u(x_4,...,x_5) - будь-яка нелінійно залежна від всіх змінних x_4,...,x_n булева функція з якомога більшим спепенем d(u (x_4,...,x_n)) і значенням нелінійності N(u(x_4,...,x_n)) - це забезпечує високі значення зазначених параметрів для результуючої булевої функції f(x_1,...,x_n). Булева функція e(x_3,...,x_n) - лінійна булева обов'язково залежна від x_3. Тоді, згідно з доведеною теоремою булева функція g(x_3,...,x_n) буде балансною функцією, оскільки є сумою балансної функції і функції не залежної від x_3. При цьому функція g(x_3,...,x_n) не є SAC оскільки змінна x_3 входить лінійно в g(x_3,...,x_n).

Булева функція h(x_3,...,x_n), що відповідає критерію строго лавинного ефекту тобто є SAC-функцією і приймаюча середнє арифметичне значення рівно на 2^{n-4} наборах і така, що g(x_3,...,x_n) · h(x_3,...,x_n) = h(x_3,...,x_n) и g(x_3,...,x_n) · h($x_3 \oplus 1$,...,$x_n \oplus 1$)= 0 може бути знайдена за способом, викладеному в попередньому розділі.

В якості ілюстрації пропонованого формалізованого методу отримання для криптографічних алгоритмів булевих функцій, відповідних критерію максимуму повної та умовної ентропії (тобто балансній SAC-функції) в таблиці 2.2. наведено приклад отримання балансної SAC - функції від 6 змінних.

Таблиця 2.2.

SAC - функції від 6 змінних

$x_1\ x_2\ x_3\ x_4$	g	h	f_1	f_2	f_3	f_4	$x_3\ x_4\ x_5\ x_6$	g	h	f_1	f_2	f_3	f_4
0 0 0 0	0	0	1	1	0	1	1 0 0 0	0	0	1	0	0	0
0 0 0 1	0	0	1	0	0	0	1 0 0 1	1	1	1	0	1	1
0 0 1 0	0	0	1	0	0	0	1 0 1 0	0	0	1	1	0	1
0 0 1 1	1	1	1	0	1	1	1 0 1 1	1	0	0	0	0	1
0 1 0 0	0	0	1	0	0	0	1 1 0 0	0	0	1	1	0	1
0 1 0 1	1	1	1	0	1	1	1 1 0 1	1	0	0	0	0	1
0 1 1 0	0	0	1	1	0	1	1 1 1 0	1	0	0	0	0	1
0 1 1 1	0	0	0	0	0	1	1 1 1 1	1	1	1	0	1	1

Підстановка функцій f_1, f_2, f_3, f_4 в формулу (2.5) дозволяє отримати результуючу булеву функцію $f(x_1,...,x_6)$ у вигляді : $f(x_1,...,x_6)=1 \oplus x_1 \oplus x_2x_3 \oplus x_2x_4 \oplus x_2x_5 \oplus x_2x_6 \oplus x_1x_2 \oplus x_4x_5x_6 \oplus x_3x_5x_6 \oplus x_3x_4x_5 \oplus x_3x_4x_6 \oplus x_2x_3x_6 \oplus x_2x_4x_6 \oplus x_2x_5x_6 \oplus x_1x_5x_6 \oplus x_1x_4x_6 \oplus x_1x_3x_6 \oplus x_1x_3x_4x_5 \oplus x_2x_4x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_2x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_1x_4x_5x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6$.

Отримана функція є балансною, відповідає критерію SAC і SAC(1), її нелінійність дорівнює 16.

Основною перевагою запропонованого методу є простота і відносно невеликі витрати часу в порівнянні з відомими методами і, зокрема в порівнянні з находженням SAC-функцій використовуючи зворотне перетворення Уолша.

Крім того, запропонований метод в порівнянні зі згаданим методом отримання булевих функцій, відповідних критерію лавинності, з використанням Уолш-перетворень має розширені функціональні можливості, оскільки він забезпечує отримання балансної результуючої функції, що не досягається при використанні методу зворотного Уолш-перетворення.

ВИСНОВКИ ДО РОЗДІЛУ 2

Проведені в рамках цього розділу дослідження, спрямовані на створення ефективного методу отримання булевих функцій, що володіють лавинним ефектом (максимумом диференціальної ентропії) і високою нелінійністю можна зробити наступні висновки:

1. Теоретично обґрунтований новий підхід отримання булевих функцій, що мають лавинний ефект і високу нелінійність, які визначають здатність криптографічних алгоритмів протистояти лінійному і диференціальному криптоанализу. Підхід полягає в направленій модифікації довільної балансної булевої функції шляхом перестановки її одиничних значень в таблиці істинності для отримання лавинного ефекту.
2. Теоретично обґрунтований, розроблений і досліджений метод побудови булевих функцій зазначеного класу, відмінністю якого є мала трудомісткість і простота, а також те, що він дозволяє отримувати балансні SAC-функції, тобто булеві функції, що володіють максимумом повної і умовної ентропії. Для експериментального дослідження запропонованого методу написані програми, які практично апробовані і їх практичне використання підтвердило справедливість теоретичних передумов, закладених в їх основу.
3. Розроблена та практично реалізована у вигляді програмного продукту модифікація розробленого методу синтезу булевих функцій спеціальних класів, відмінністю якої є те, що вона оперує не з таблицями істинності, а з алгебраїчною нормальною формою функцій, що зменшує вимоги до ресурсів пам'яті і дозволяє будувати булеві функції, що володіють специфічними властивостями, важливими для криптографічних застосувань від великого числа змінних.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМ СИНТЕЗУ БУЛЕВИХ ФУНКЦІЙ, ЩО ЗАДОВОЛЬНЯЮТЬ SAC

3.1. Розробка програми синтезу булевих SAC функцій з використанням таблиць істинності.

В інтерактивному варіанті простановка одиниць виконується оператором в тих місцях, де це можливо з автоматичним блокуванням проставляння одиниць в заборонених позиціях таблиці істинності. Програма була розроблена для дослідження оптимального за критерієм досягнення максимальної нелінійності генерованої булевої SAC-функції плану проставляння одиниць в таблиці істинності.

Для зберігання таблиці істинності використовується байтовий масив F, а в двовимірному масиві X організовано зберігання наборів значень вхідних змінних.

Виконання програми починається з того, що генерується заповнення двовимірного масиву X. Для цього виконується сканування змінної i від 1 до m, що дорівнює 2^n, де n-число вхідних змінних на яких визначена генерована булева функція. Для кожного значення i здійснюється сканування змінної j від 1 до n і в циклі сканування цієї змінної виконується переклад проміжної змінної h, початкове значення якої дорівнює i-1 в двійкову систему числення і отриманий вектор двійкового представлення є i-тий рядок матриці X.

Потім здійснюється вивід поточного значення матриці X і відповідного значення функції F у вигляді таблиці, причому невизначені значення функції виводяться на екран символом 'x'. Після введення оператором номера набору, на якому створювана функція приймає одиничне значення в змінну h, реалізується копіювання в вектор Y рядка з номером h матриці X.

Потім починається цикл з використанням змінної i, яка змінюється від 1 до m для перебору всіх рядків матриці X. Для кожного поточного рядка матриці X виконується обчислення хеммінгової відстані між поточним рядком матриці X і вектором Y, для цього організовується внутрішній цикл по змінній j яка змінюється від 1 до n і в якому зіставляються j-ті компоненти вектора Y і рядки матриці X. В разі розбіжності, лічильник хеммінгової відстані h збільшується на одиницю.

Таким чином знаходиться набір, що співпадає із заданим, значення функції на ньому встановлюється рівним одиниці. Крім того, знаходяться всі набори, на яких рядок матриці X відрізняється від вектора Y тільки в одній позиції (тобто хеммінгова відстань дорівнює одиниці) і значення функції на цьому наборі проставляється рівним нулю. У разі якщо на згаданих наборах значення функції вже було визначено раніше, то видається повідомлення про помилковий вибір оператором набору на якому він проставив середнє арифметичне значення.

Описаний цикл триває до тих пір, поки в таблиці істинності не будуть проставлені всі 2^{n-2} одиниць. Після закінчення введення виконується переклад побудованої функції з табличного представлення в алгебраїчну нормальну форму.

Для побудови алгебраїчної нормальної функції використовується рекурсивна функція transform, яка формує числове стисле представлення термів при заданому n-вимірному векторі Y - який представляє собою набір на якому досліджувана функція приймає одиничне значення.

Результатом роботи рекурсивної функції transform є множина двійкових чисел які відповідають набору Y.

Наприклад, якщо набір Y = {011010} то набір двійкових чисел, відповідний до цього набору буде:

011010 = 22

111010 = 23

011110 = 30

011011 = 52

111110 = 31

111011 = 53

011111 = 62

111111 = 63

Дійсно, набір { 0 1 1 0 1 0 } відповідає конституенті 1, що рівна: (1 ⊕ x1) x2 x3 (1 ⊕ x4) x5 (1⊕ x6) = x1x2x5 ⊕ x1x2x3x5 ⊕ x4x2x3x5 ⊕

⊕ x2x3x5x6 ⊕ x1x2x3x4x5 ⊕ x1x2x3x6x5 ⊕ x6x2x3x4x5 ⊕ x1x2x3x4x5x6

і наведені вище числа, що записуються в масив представляють відповідні з термів на які розкладається розглянута конституента 1.

Для побудови сукупності термів конституенти 1 використовується рекурсивна функція transform. Параметр і цієї функції вказує номер розряду який аналізується для побудови сукупності термів. При першому зверненні до функції transform параметр і встановлюється рівним 0. Якщо перша змінна x1 входить в набір, на якому функція приймає одиничне значення то вага розряду, відповідного цій змінній додається до поточного значення allExchanges, інакше починається формування нового числа в яке записується вага розряду досліджуваної змінної. Якщо досліджуваний розряд не є першим, то виконується повторний виклик функції transform зі збільшеним значенням і, причому якщо Y [i] = 1 то до змінної е додається вага аналізованого розряду.

В основній програмі виконується формування списку двійкових представлень термів, які представляють собою розкладання досліджуваної функції. Покажчиком голови списку виступає змінна h, що спочатку встановлюється

рівною 0 і інкременується при додаванні в список нового терма. Потім виконується цикл сканування табличних значень досліджуваної функції - якщо функція дорівнює 0 - то триває сканування, а якщо функція дорівнює 1 то виконується копіювання набору X у допоміжний набір Y, змінна e встановлюється рівною 0 і виконується виклик описаної вище рекурсивної функції transform.

Після закінчення формування списку термів в масиві allExchanges (терми при цьому представляються для економії у вигляді двійкових чисел) необхідно виконати стиснення масиву allExchanges. Для цього використовується допоміжний масив newSeq в який копіюються неповторювані значення термів. Сам процес виключення виконується у вигляді двох вкладених циклів в першому з яких виконується сканування масиву allExchanges, а в другому порівняння його з обраним раніше елементом allExchanges при цьому виконується підрахунок знайдених в масиві allExchanges термів, що збігаються зі сканованим елементом масиву newSeq. Якщо число знайдених однакових термів парне, то в алгебраїчній формі вони взаємно знищуються - практично це означає, що вони не переносяться в масив newSeq - масив відібраних двійкових представлень термів.

Після описаного відбору неповторюваних двійкових представлень термів які увійдуть в алгебраїчну нормальну форму виконується їх сортування за критерієм неспадання числа одиниць в двійковому поданні. Це робиться для зручності подальшого аналізу алгебраїчної нормальної форми - вона виявляється впорядкованою: спочатку терми малої кратності - потім терми більшої кратності.

У завершенні процесу формування алгебраїчної нормальної форми виконується перетворення з числової форми, яка при формуванні була більш компактною і зручною, в строкову, яка більше зручна для візуального аналізу.

Якщо в прикладних наукових програмах дослідження булевих функцій проводиться подальший машинний аналіз отриманої алгебраїчної нормальної форми, то використовується її двійкове подання з використанням масиву newSeq.

При перетворенні числового запису сукупності термів, що зберігаються в масиві newSeq в строкову проводиться роздільне перетворення кожного терма - елемента масиву newSeq. Для цього виконується цикл перебору від 1 до h всіх елементів масиву newSeq - які відповідають термам представлення функції в алгебраїчній нормальній формі. При аналізі кожного елемента масиву newSeq виконується переклад цілого числа в двійкову систему числення і якщо і-тий розряд цього подання дорівнює 1, то до рядка терма конкатенуються рядок "xi". Результуючий рядок виводиться на екран дисплея і записується в текстовий файл результату.

3.2. Розробка програм синтезу булевих SAC-функцій з використанням алгебраїчної нормальної форми

Програма генерації SAC-функцій за методом розкладання на систему балансних підфункцій призначена для отримання алгебраїчної нормальної форми булевих функцій, що задовольняють критерію строгого лавинного ефекту, тобто SAC-критерію.

У розробленій програмі використовуються змінні типу binaryInfo для збереження терма алгебраїчної нормальної форми, змінна зазначеного типу містить байтовий масив A, число компонент якого відповідає максимальному числу amount змінних, що задається користувачем, причому елементи цього масиву можуть приймати тільки два значення: одиниця на і-тій позиції, якщо і-та змінна входить в описуваний терм і нуль - якщо ця змінна не входить в терм.

До складу розробленої програми входять наступні функції. Функція getBestTables - перевірки згенерованої булевої функції на відповідність лавинному критерію по всім змінним.

Вхідними параметрами функції є table функція і amount - кількість всіляких наборів на яких досліджувана функція приймає значення. Усередині функції використовується прапор ideal відповідності SAC-критерію: спочатку функції він встановлюється в одиницю і якщо в ході перевірок виявиться, що одна з умов відповідності SAC-критерію не виконується, то цей прапор встановлюється в нуль. У самій функції організовується спочатку цикл зовнішнього перебору в якому перебираються всі n змінні. Усередині зовнішнього циклу формується допоміжний m-мірний вектор allExchanges, всі елементи якого спочатку встановлюються рівними нулю. Потім організовується внутрішній цикл, в якому виконується сканування всіх можливих наборів з n змінними при цьому на початку внутрішнього циклу виконується копіювання вектора набору у допоміжний вектор result в якому інвертується один розряд, а саме той, який відповідає активній змінній. Після цього знову виконується цикл сканування всіх можливих наборів n булевих змінних і вектори наборів порівнюються зі зміненим вектором result до знаходження такого набору, який би співпадав з вектором result - таким чином знаходяться два значення функції на наборах, що відрізняються інверсним і прямим значенням активної змінної - відповідно, ці два значення порівнюються і якщо вони не рівні, то виконується інкремент лічильника числа розбіжностей. Таким чином, після закінчення внутрішнього циклу фактично на лічильнику числа розбіжностей буде зафіксований код числа наборів n-1 змінних для яких виявлено розбіжність, який порівнюється з числом наборів для яких виявлено збіг значень функції при інвертуванні активного розряду - якщо при цьому не буде виявлено рівність - це означає що для активної змінної критерій SAC не виконується і відповідно прапор

ideal скидається в нуль. Значення result повертається функцією в основну програму в якості результату роботи функцією getDerivs.

Функція getDerivs - яка виконує обчислення фактичного значення нелінійності функції, код якої повертається в формальній змінній derivs. Визначення нелінійності виконується за допомогою пошуку мінімуму хеммінгової відстані досліджуваної булевої функції та всіх лінійних функцій. Для цього організовується перебір від 1 до 2^n-1 всіх лінійних булевих функцій. Всередині циклу за номером k булевої функції здійснюється генерація лінійної булевої функції за наступним алгоритмом: індексу k ставиться у відповідність двійковий вектор Y = (y1,y2, ..., yn) який представляє собою двійкове подання цілого числа k - якщо yi = 1 то змінна xi входить в нормальну алгебраїчну форму k-тій лінійної булевої функції. Далі всередині основного циклу перебору всіх лінійних функцій виконується цикл перебору всіх наборів з n двійкових змінних - для кожного такого набору обчислюється значення R лінійної булевої функції за формулою:

$$R = \left(\sum_{i=1,..,n} x_i y_i \right) mod\ 2 \qquad (3.1)$$

Після обчислення значення R лінійної булевої функції воно порівнюється зі значенням seq досліджуваної функції на цьому наборі - якщо значення не збігаються, то інкрементується величина поточної хеммінгової відстані.

Таким чином організовується обчислення хеммінгової відстані між досліджуваної булевої функцією і сканованою лінійною - D. Обчислена таким чином хеммінгова відстань порівнюється з поточним мінімальним (початкове значення поточного мінімального встановлюється рівним 10000). Таким чином після закінчення зовнішнього циклу перебору всіх лінійних функцій від n змінних буде знайдена мінімальна хеммінгова відстань між досліджуваною

булевої функцією і найближчою лінійною функцією.

Функція compare призначена для сортування термів алгебраїчної нормальної форми в її кінцевому запису так, щоб у підсумковому запису АНФ спочатку йшли короткі терми, а потім - довгі. Це забезпечує кращі умови для візуального аналізу АНФ в процесі досліджень. Процес сортування реалізований з використанням внутрішніх функцій мови javaScript.

Функція getInfo виконує виведення алгебраїчної нормальної форми згенерованої булевої функції на екран.

Функція getTable здійснює вилучення часткової функції result з булевої функції seq, причому в створюваний ланцюжок result відбираються тільки ті терми, які містять змінну a. У функції реалізується сканування ланцюжка seq і для кожної поточної її змінної перевіряється значення a-того елемента масиву result. Якщо зазначений елемент дорівнює одиниці, то змінна дописується в список seq, причому значення a-того елемента масиву result цієї динамічної змінної встановлюється в нуль.

Функція copy - реалізує копіювання масиву array.

Функція generate виконує генерацію балансної булевої функції від змінних заданих послідовністю номерів від k до m. b-номер балансної змінної, яка лінійно входить в генеровану функцію і таким чином забезпечує її балансність. Процедура генерує початковий терм алгебраїчної нормальної форми у вигляді вектора, що містить всі нульові компоненти, крім компонента з номером b.

Потім формується випадкове число термів, кожен з яких містить згенеровані випадковим чином одиниці або нулі на компонентах вектора A з номерами від k-го до m-го. Сформовані таким чином терми перевіряються на неповторюванність (шляхом порівняння з усіма згенеруваними раніше) і дописуються до списку генерованої балансної булевої функції з покажчиком W.

В основній програмі здійснюється введення числа n змінних на яких визначена будована булева функція, відкриття текстового файлу результатів, потім реалізується генерація першої балансової функції системи, яка виконується функцією generate.

При цьому з використанням вбудованого генератора виконується розбиття множини всіх змінних на дві непустих підмножини. Спочатку генеруються балансові функції для змінних приналежних першій підмножині. Для цього на другій множині випадковим чином вибирається балансова змінна і викликається функція generate, яка генерує балансну булеву функцію, домножає все її терми на активну змінну і дописує до списку повної функції.

Аналогічні операції виконуються і для змінних друї підмножини: також виконується їх перебір з виділенням активної змінної, потім для активної змінної випадково вибирається балансна змінна з елементів першої множини змінних і для них будується балансна функція, визначена на першій множині змінних, отримана функція домножається на активну змінну і дописується до списку генерованої булевої функції.

Із завершенням формування алгебраїчної нормальної форми здійснюється перевірка її на відповідність критерію SAC, обчислення нелінійності і виведення у вихідний текстовий файл. У програмі передбачений режим автоматичного генерування SAC-функцій з фільтрацією їх по гранично-допустимому мінімальному значенню нелінійності.

ВИСНОВКИ ДО РОЗДІЛУ 3

В результаті досліджень, що складають цей розділ, і спрямовані на створення програмних засобів побудови булевих функцій, що мають специфічні властивості, що визначають їх ефективне застосування в криптографічних алгоритмах захисту інформації можна зробити наступні висновки:

1. Розроблено й апробовано практичним використанням комплекс програм для автоматизованої побудови булевих SAC-функцій від великого числа змінних, який може бути практично використаний для проектування алгоритмів криптографічного захисту інформації в комп'ютерних інтегрованих системах, базах даних і мережах.
2. З використанням розроблених програмних засобів проведено порівняльний аналіз запропонованого методу побудови SAC-функції з відомими. Такі експериментальні дослідження показали наявність позитивного ефекту, зокрема, запропонований метод має більшу продуктивність, забезпечує генерацію SAC-функцій високих порядків і не накладає обмежень на число змінних.
3. Розроблені програмні засоби можуть бути практично використані для побудови нових і модифікації існуючих алгоритмів криптографічного захисту інформації та, в першу чергу, алгоритмів симетричного шифрування, що використовують підстановочні блоки, а також алгоритмів потокового шифрування даних.

ВИСНОВКИ

В результаті виконання цієї бакалаврської роботи, спрямованої на створення методів синтезу булевих функціональних перетворень спеціальних класів для систем криптографічного захисту інформації, а також програмних засобів для реалізації таких методів були отримані наступні результати:

1. Виконано огляд методів криптографічних алгоритмів, що застосовуються в сучасних системах захисту інформації в комп'ютерних системах і мережах, проведений оглядовий аналіз використання методів функціонального аналізу для визначення стійкості до розкриття широкого класу криптографічних алгоритмів. В результаті аналізу показано, що стійкість широкого класу криптографічних алгоритмів, в основі яких важковирішувана математична задача відшукання коренів систем нелінійних булевих функцій, визначається відповідністю булевих функцій бітового перетворення певним критеріям, одним з яких є критерій максимуму умовної ентропії або строгого лавинного ефекту. Показано, що задача синтезу таких функцій для створення криптостійких алгоритмів захисту інформації є далеко не тривіальним завданням.
2. Виконано оглядовий аналіз існуючих методів отримання булевих функцій зазначеного класу, який дозволив виявити їх основні недоліки: велику трудомісткість і непридатність для синтезу функцій від великого числа змінних, і таким чином обгрунтована необхідність продовження робіт в цьому напрямку.
3. Теоретично обґрунтований, розроблений і досліджений метод побудови булевих функцій зазначеного класу, відмінністю якого є мала трудомісткість і прототa, а також те, що він дозволяє отримувати балансні SAC-функції, тобто булеві функції, що володіють максимумом повної і умовної ентропії. Для експериментального дослідження

запропонованого методу написані програми, які практично апробовані і їх практичне використання підтвердило справедливість теоретичних передумов, закладених в їх основу.

4. Розроблено й апробовано практичним використанням комплекс програм для автоматизованої побудови булевих SAC-функцій від великого числа змінних, який може бути практично використаний для проектування алгоритмів криптографічного захисту інформації в комп'ютерних інтегрованих системах, базах даних і мережах.
5. Проведено порівняльний аналіз запропонованого методу побудови SAC-функцій з відомими, який показав наявність позитивного ефекту, зокрема, запропонований метод має більшу продуктивність, забезпечує генерацію SAC-функцій високих порядків і не накладає обмежень на число змінних.
6. В цілому, проведені в рамках цієї бакалаврської роботи дослідження підтвердили в повній мірі перспективність робіт, спрямованих на створення методів і засобів автоматизованого проектування систем булевих функцій, які володіють специфічними властивостями для криптографічних алгоритмів захисту інформації в комп'ютерних системах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Carlet C. Boolean functions for cryptography and error-correcting codes./ C/ Cerlet, Y. Grama, R. Hammer // Eds.: Cambridge University Press.UK. – 2017.- 277 p.
2. Mesnager S. Bent Functions: Fundamentals and Results. NY. Springer-Verlag.- 2016.- 167 p.
3. Li M. Sequences, Adjacency , Graph and Cyclotrons / Li M., Lin D. //IEEE Transaction on Information Theory,-2018,- Vol.64, № 4, - P. 2942-2953.
4. Xiang C. A construction of Linear Codes from Boolean Functions / C. Xiang, Feng K., Taug C. // IEEE Transaction on Information Theory,-2017,- Vol.63, № 1 - P. 167-176.
5. Gouget A. On the propagation criterion of Boolean functions / A. Gouget //. Proceedings of the Workshop on Coding, Cryptography and Combinatorics 2003, Birkh¨auser Verlag, - 2004.- PP.153-168.
6. Hawkes P. Rewriting Variables: The Complexity of Fast Algebraic Attacks on Stream Ciphers. / P. Hawkes and G. Rose. // Proceedings of CRYPTO 2004, Lecture Notes in Computer Science 3152, - 2004.- PP. 390–406.
7. Hu H. On quadratic bent functions in polynomial forms / H. Hu and D. Feng. // IEEE Transaction on Information Theory,-2018,- Vol.64, № 4, - PP. 2932-2945.
8. Kurosawa K. Almost security of cryptographic Boolean functions/ K. Kurosawa and R. Matsumoto // IEEE Transaction on Information Theory,-2014,- Vol.60, № 11, - PP. 2752-2761.
9. Марковский А.П. Получение булевых преобразований специальных классов для построения эффективных алгоритмов защиты информации / Марковский А.П., Зюзя А.А., Шерстюк В.Д. // Вісник Національного технічного університету України ”КПІ”. Інформатика, управління та обчислювальна техніка. К.,"ВЕК++",- 2008.- № 49.- С.7-13.

10. Марковский А.П. Получение систем ортогональных булевых SAC–функций для систем защиты информации. / А.П.Марковский, Абу Усбах А.Н., Аль-Омар Салех.//Вісник Національного технічного університету України ”КПІ”. Інформатика, управління та обчислювальна техніка. 2001, - № 36. - С.94-108.
11. Марковский А.П. Получение балансных булевых SAC–функций для систем защиты информации / А.П. Марковский, В.В. Осадчий, Аль-Омар Салех //Вісник Національного технічного університету України ”КПІ”. Інформатика, управління та обчислювальна техніка.-2001.- № 36.-С.54-60.
12. Lacharme P. Nonlinearity of some invariant Boolean functions./ P. Langevin and J.-P. Zanotti.//Designs, Codes and Cryptography Vol. 48, - 2016.- PP. pp. 131 - 146,
13. Николайчук Я.М. Коди полів Галуа: теорія і застосування./ Я.В.Николайчук // Тернопіль.-Вид-во ТНУ.-2012.- 576 с.
14. Li N. Symmetric Boolean functions depending on an odd number of variables with maximum algebraic immunity / N.Li and W.Qi.. // IEEE Transaction on Information Theory,-2018,- Vol.63, № 5, - PP. 2271-2273.
15. Li N. On the construction of Boolean functions with optimal algebraic immunity/ N. Li, L. Qu, W.-F. Qi, G. Feng, C. Li and D. Xie. // IEEE Transaction on Information Theory,-2008,- Vol.53, № 3, - PP. 1330-1334.
16. Lacharme P. On the nonlinearity of power functions./ P. Langevin and P. V´eron. // Designs, Codes and Cryptography. Vol. 47, 2015. - PP. 31 - 43,
17. Meng Q. On the degree of homogeneous bent functions / Q. Meng, H. Zhang, M. Yang and J. Cui. //. Discrete Applied Mathematics Volume 165, № 5. - 2017.- PP. 665-669.
18. Qu L. Constructing symmetric Boolean functions with maximum algebraic immunity / L. Qu, K. Feng, L. Feng and L. Wang // IEEE Trans. on Inf. Theory, vol. 55- № 6.- 2009.- PP. 2406-2412.

19. Mesnager S. Improving the lower bound on the higher order nonlinearity of Boolean functions with prescribed algebraic immunity / S. Mesnager // IEEE Transactions on Information Theory, vol. 54, no. 8, pp. 3656 - 3662, 2008.- PP.3656-3662.
20. Алферов А.П., Основы криптографии: Учебное пособие / А.П. Алферов А.Ю. Зубов, А.С., Кузьмин, А.В., Черемушкин // 2-е изд., испр. и доп. – М.: Гелиос АРВ, 2002. – 480 с. ил.
21. Wang W. On the Relation Between Identifiability , differential Privacy and Mutual Informational Privacy / Wang W., Ying I., Zhang J. // IEEE Trans. Inf. Theory.- Vol. 62.- 2016.- № 9.- P. 5018-5029.
22. Харин Ю.С. и др. Математические основы криптологии: Учеб. пособие / Ю.С. Харин, В.И. Берник, Г.В. Матвеев. – //Мн.: БГУ, 1999.- 319с.: ил.
23. Самофалов К.Г., Метод получения булевых балансных SAC-функций для систем защиты информации. / К.Г.Самофалов, А.П.Марковский, Гаваагийн Улзисайхан, Бардис Н., // Вісник Національного технічного університету України "КПІ". Інформатика, управління та обчислювальна техніка.-1998.- № 31.- С.131-140.
24. Самофалов К.Г. Использование аппарата булевых функций для оценки эффективности криптографических алгоритмов защиты информации/ Самофалов К.Г., Эль-Хами И., Кожемякин С.В. // Збірник статей "Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні".- К:,Вид-во ЕКМО, 2000.- С.244-250.

ДОДАТОК 1

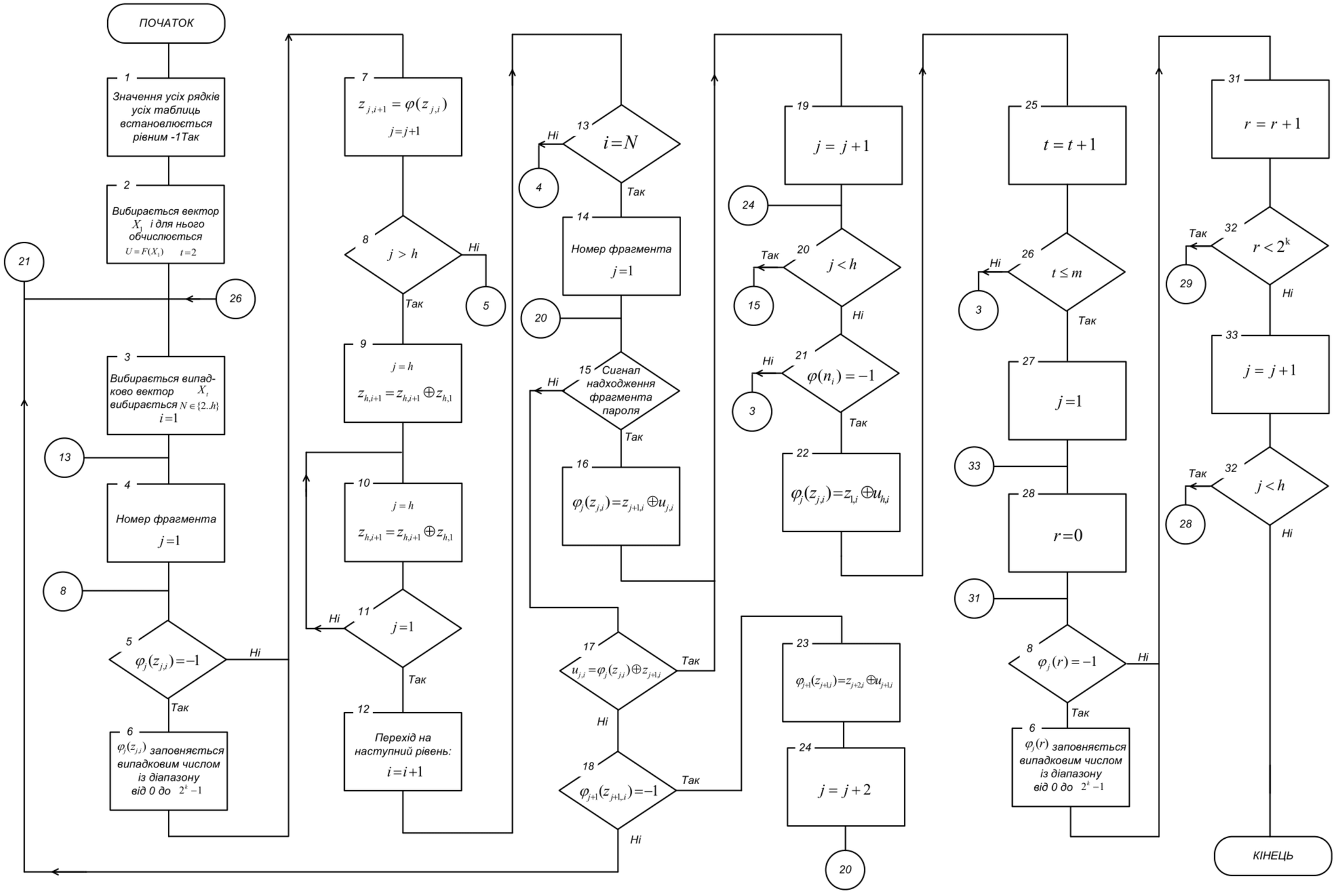
Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації

Принципова схема

ІАЛЦ.467200.004 Д1

Аркушів 1

Київ 2020 р.



Зм.	Лист	№докум.	Підп.	Дата	ІАЛЦ.467200.004 Д1		
Розроб.		Буровецька К.О.			Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації Принципова схема	Літ.	Маса	Масштаб
Перев.		Марковский О.П.				Д		
Н.контр.		Сімоненко В.П.				Аркуш	Аркушів 1	
Затв.		Марковский О.П.				НТУУ «КПІ ім. Ігоря Сікорського» ФІОТ гр.ІО-64		

ДОДАТОК 2

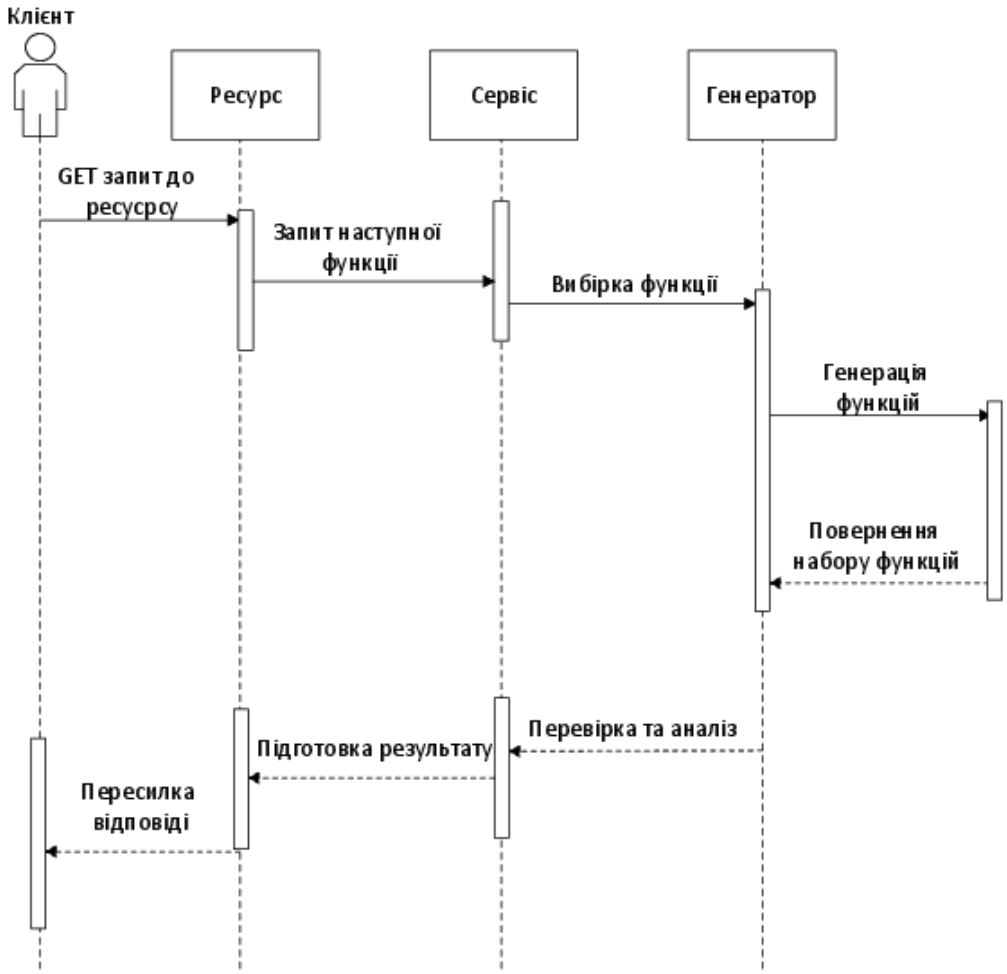
Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації

Структурна схема пристрою

ІАЛЦ.467200.005 Д2

Аркушів 1

Київ 2020р.



					ІАЛЦ.467200.005 Д2	
Зм.		*№ документа*	*Підпис*	*Дата*		
Розробила		*Буровецька К.О*			Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації Структурна схема	*Літ.* | *Аркуш* | *Аркушів* *1* | *1*
Перевірив		*Марковський О.П*				
Н. Контр.		*Сімоненко В.П.*				*НТУУ «КПІ ім. Ігоря Сікорського» ФІОТ гр. ІО-64*
Затвердив		*Стіренко* С.Г.				

ДОДАТОК 3

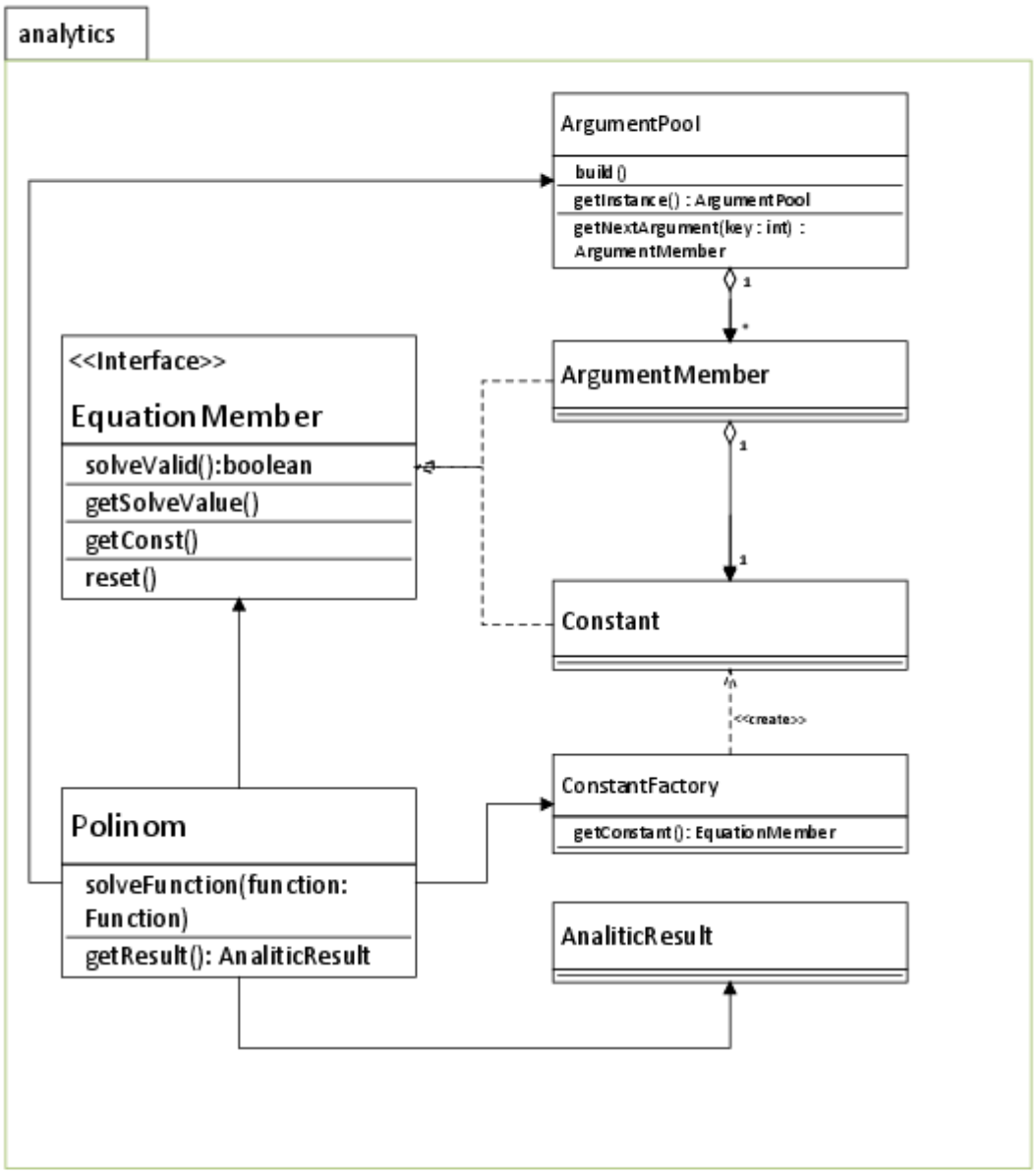
Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації

Функціональна схема алгоритму

ІАЛЦ.467200.006 Д3

Аркушів 1

Київ 2020 р.



					ІАЛЦ.467200.006 ДЗ	
Зм.		*№ документа*	*Підпис*	*Дата*		
Розробила		*Буровецька К.О*			Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації Структурна схема	*Літ.* / *Аркуш* / *Аркушів*: 1 / 1
Перевірив		*Марковський О.П*				
						НТУУ «КПІ ім. Ігоря Сікорського» ФІОТ гр. ІО-64
Н. Контр.		*Сімоненко В.П.*				
Затвердив		*Стіренко С.Г.*				

ДОДАТОК 4

Метод та програмні засоби побудови булевих перетворень для криптографічних алгоритмів захисту інформації

Лістинг програми синтезу булевих функцій

ІАЛЦ.467200.007 Д4

Аркушів 8

Київ 2020

```
<!DOCTYPE html>
<html lang="en" ng-app="science">
    <head>
    <meta charset="UTF-8">
    <title>Title</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script src="angular.min.js"></script>
    <script src="science.js"></script>
    <script src="sequence.js"></script>
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="style.css">
    </head>
    <body ng-controller="MainController" ng-keypress="invertDialog($event);
unfix($event)">
    <div>
    <div ng-show="active">
    <div class="col-md-1 col-lg-1 col-sm-1">
    <button class="btn btn-primary" ng-click="previous()">Previous</button>
    <button class="btn btn-primary" ng-click="getPath()">Path</button>
    <button class="btn btn-primary" ng-click="clear()">Clear</button>
    </div>
    <div class="col-md-3 col-lg-3 col-sm-3"
ng-repeat="a in active" ng-init="outer = $index">
    <table class="table table-hover table-bordered"
ng-repeat="current in a.arraysLevel" ng-click="tableClick($event, current.object,
outer)"
ng-mousemove="displayInfo($event)" ng-mouseleave="hideInfo()"
ng-mouseenter="getInfo(current)">
    <tr>
    <td ng-class="{info : current.object.correct}"></td>
    <th ng-repeat="i in current.object.data.columns">df{{i}}</th>
</tr>
<tr ng-repeat="row in current.object.data">
    <td>dx{{row.x}}</td>
<td ng-repeat="el in row" ng-class="{danger: el.worst, success: el.best, info:
el.ideal}">{{el.a}}</td>
</tr>
</table>
</div>
<div class="col-md-1 col-lg-1 col-sm-1">
    <button class="btn btn-primary" ng-click="next()">Next</button>
    </div>
    </div>
    <div ng-hide="active" class="container" style="margin-top: 15px">
    <input type="text" ng-model="amount">
    <button id=initButton" class="btn btn-primary" data-loading-text = "Loading..."
ng-click="init($event)">Create new sequence</button>
</div>
</div>
<div id="dialog" ng-show="showDialog" >
    <h6 ng-repeat="line in info">
    {{line.from}} -> {{line.to}}<span ng-show="line.changed">*</span>
    </h6>
    </div>
    </body>
    </html>
```

STYLE.CSS

```
table, tr, td, th{
    padding: 5px;
    text-align: center;
}
```

```
#dialog{
    position: absolute;
    background: #9acfea;
    padding:15px;
    font-family: Consolas, Monospaced;
}
```

SCIENCE.JS

```
let app = angular.module('science', []);

app.controller('MainController', ['$scope',function($scope){

    let derivLevels = [];
    let binaryInfo = [];
    let decimalInfo = [];
    let windowFixed = false;

    $scope.firstActive = 0;
    $scope.info = binaryInfo;
    $scope.showDialog = false;

    $scope.clear = function () {
        $scope.active = null;
    };

    $scope.getActive = function(/*number*/a){
        let b = a+3;
        b = b < derivLevels.length ? b : derivLevels.length;
        let active = [];
        for (let k = a; k < b; k++) {
            let arrays = [];
            let currentDerivLevel = derivLevels[k];
            for (let i = 0; i < currentDerivLevel.length; i++) {
                arrays.push({'object':currentDerivLevel[i]});
            }
            active.push({'arraysLevel':arrays});
        }
        $scope.active = active;
    };

    $scope.getPath = function(){
        $scope.path = $scope.sequence.transform($scope.seq, {"weight":[0]}, 0);
        derivLevels[0][0].correct = true;
    };

    $scope.invertDialog = function($event){

        if($event.charCode == ($event.DOM_VK_B + (98 - 66))) {
            $scope.info= binaryInfo;
        }
        if($event.charCode == ($event.DOM_VK_D + (98 - 66))) {
            $scope.info= decimalInfo;
        }
    };

    $scope.previous = function(){
        if(this.firstActive == 0)
            alert("Reach start");
        else {
            this.firstActive--;
            this.getActive(this.firstActive);
        }
```

```
};

$scope.next = function(){
    if(this.firstActive == derivLevels.length-3)
        alert("Reach end");
    else {
        this.firstActive++;
        this.getActive(this.firstActive);
    }
};

$scope.tableClick = function($event, table, outer){
    if(!$event.ctrlKey) {
        let index = $scope.firstActive + outer + 1;
        derivLevels.length = index;
        let tables = $scope.sequence.getBestTables(table, 10);
        if (table.correct) {
            if (index - 1 < $scope.path.length) {

                for (let i = 0; i < tables.length; i++) {

                    if (tables[i].from == $scope.path[index - 1].from &&
                        tables[i].to == $scope.path[index - 1].to) {
                        tables[i].correct = true;
                        break;
                    }
                }
            }
        }
        derivLevels[$scope.firstActive + outer + 1] = tables;
        derivLevels[$scope.firstActive + outer] = [table];

        if (outer == 2)
            $scope.firstActive++;
        $scope.getActive($scope.firstActive);
    }
    else{
        windowFixed = !windowFixed;
    }
};

$scope.init = function ($event) {
    $scope.sequence = new Sequence($scope.amount);
    let sequence = $scope.sequence;
    let seq = sequence.generate();
    let table = sequence.getTable(seq);
    $scope.seq = seq;
    derivLevels = [];
    derivLevels[0] = [table];
    $scope.firstActive = 0;
    $scope.getActive(0);
};

function format(string, length, filler){
    let str = string;
    while(str.length < length)
        str = filler + str;
    return str;
}

$scope.getInfo = function(object){
    let table = object.object;
    let prevSeq = table.parent && table.parent.state;
    let currentSeq = table.state;
```

```
        binaryInfo.length = 0;
        decimalInfo.length = 0;

        for(let i = 0 ; i < currentSeq.length; i++){
            binaryInfo[i] = {};
            decimalInfo[i] = {};

            binaryInfo[i].to = format(currentSeq[i].toString(2), $scope.amount, '0');
            decimalInfo[i].to = format(""+currentSeq[i], 4, '\u00A0');

            if(prevSeq){

                binaryInfo[i].from = format(prevSeq[i].toString(2), $scope.amount,
'0');
                decimalInfo[i].from = format(""+prevSeq[i], 4, '\u00A0');
                binaryInfo[i].changed = binaryInfo[i].from != binaryInfo[i].to;
                decimalInfo[i].changed = decimalInfo[i].from != decimalInfo[i].to;
            }
            else{
                binaryInfo[i].from = '';
                decimalInfo[i].from = '';
            }
        }
    };

    $scope.displayInfo = function($event){
        if(!windowFixed){
            let a = document.getElementById("dialog");
            a.style.left = $event.pageX+10 + "px";
            a.style.top = $event.pageY+10 + "px";
        }
        $scope.showDialog = true;
    };

    $scope.hideInfo = function(){
        if(!windowFixed) {
            $scope.showDialog = false;
        }
    };

    $scope.unfix = function($event){
        if($event.charCode == ($event.DOM_VK_F + (98 - 66))) {
            windowFixed = false;
            $scope.hideInfo();
        }
    }
}]);
```

SEQUENSE.JS

```
function Sequence(/*number*/sz) {
    let size = sz;
    let ideal = 1 << (sz-2);
    let length = 1 << size;
    let maxPossibleWays = getPossibleWays();

    this.generate = function () {
        let size = length;
        let array = [];

        for (let i = 0; i < size; i++) {
            array[i] = i;
        }
        let resArr = [];
```

```
    for (let i = 0; i < size; i++) {
        let rand = getRandomNumber(array.length);
        resArr.push(array[rand]);
        array.splice(rand, 1);
    }
    return resArr;
};

function getRandomNumber(max) {
    let r = Math.random();
    r = max * r;
    return Math.floor(r);
}

this.toString = function(array) {
    let string = "";
    for (let i = 0; i < array.length; i++) {
        let str = array[i].toString(2);
        while (str.length < size)
            str = "0" + str;
        string += str + "    " + array[i].toString() + "\n";
    }
    return string;
};

function getWeight(/*[][]*/deriv){
    let res = [];
    let length = ideal/2+1;
    for(let i = 0; i < length; i++)
        res[i] = 0;

    for(let i = 0; i < deriv.length; i++){
        for(let j = 0; j < deriv.length; j++){
            let diff = Math.abs(ideal - deriv[i][j]);
            res[diff/2]++;
        }
    }
    return res;
}

this.getTable = function(seq){
    let result = {};
    result.parent = null;
    let deriv = this.getDerivs(seq);
    let array = [];
    let columns = [];
    for(let k = 0; k < deriv.length; k++){
        array[k] = [];
        array[k].x = k;
        columns[k] = deriv.length - k - 1;
        for(let j = 0; j < deriv.length; j++){
            array[k][j] = {'a':deriv[k][j]};
            if(deriv[k][j] === ideal)
                array[k][j].ideal = true;
        }
    }
    result.data = array;
    result.data.columns = columns;
    result.state = seq;
    return result;
};

this.getBestTables = function(table, amount){
    //масив, що містить результат обміну
```

```
        let allExchanges = [];
        //перебір перестановок
        let b = 0; //звідки
        let a = 0; //куди
        for(let i = 0; i < maxPossibleWays; i++){

            b++;
            if( b === length){
                a++;
                b = a+1;
            }

            //здійнення обміну знаходження таблиці та ваги
            let weight = getWeight(this.getDerivs(this.exchange(table.state, a, b)));
            let exc = {'from':a, 'to':b};
            allExchanges[i] = {"weight":weight, "exchange":exc};
        }

        allExchanges.sort(compare);

        let result = [];
        for(let i = 0; i < amount; i++){
            result[i] = {};
            result[i].parent = table;
            result[i].from = allExchanges[i].exchange.from;
            result[i].to = allExchanges[i].exchange.to;
            result[i].state = this.exchange(table.state, result[i].from,
result[i].to);
            let deriv = this.getDerivs(result[i].state);
            let array = [];
            let columns = [];
            for(let k = 0; k < deriv.length; k++){
                array[k] = [];
                array[k].x = k;
                columns[k] = deriv.length - k - 1;
                for(let j = 0; j < deriv.length; j++){
                    array[k][j] = {'a':deriv[k][j]};
                    if(deriv[k][j] === ideal)
                        array[k][j].ideal = true;
                    if(Math.abs(ideal - table.data[k][j].a) < Math.abs(ideal -
deriv[k][j]))
                        array[k][j].worst = true;
                    if(Math.abs(ideal - table.data[k][j].a) > Math.abs(ideal -
deriv[k][j])) {
                        array[k][j].best = true;
                        array[k][j].ideal = false;
                    }
                }
                result[i].data = array;
                result[i].data.columns = columns;
            }
        }

        return result;
    };

    this.getDerivs = function(/*[]*/seq){
        let derivs = [];

        //ініціалізація таблиці
        for(let i = 0; i < size; i++) {
            derivs[i] = [];
            for (let j = 0; j < size; j++){
                derivs[i][j] = 0;
```

```
        }
    }

    for(let k = 0; k < size; k++) {
        //дельта x
        let dx = 1 << k;
        let ddx = dx << 1;

        for (let i = 0; i < seq.length;) {
            //порівняння
            let der = seq[i]^seq[i+dx];

            //оновлення кожної DFj
            for (let j = 0; j < size; j++) {
                let mask = 1 << (size - j - 1);
                if ((der & mask) !== 0)
                    derivs[k][j]++;
            }

            if ((++i) % (ddx) === dx)
                i += dx;
        }
    }
    return derivs;
};

function copy(array) {
    let newArray = [];
    for (let i = 0; i < array.length; i++)
        newArray[i] = array[i];
    return newArray;
}

function getPossibleWays(){
    let res = 0;
    for(let i = 0; i < length; i++)
        res+= i;
    return res;
}

this.exchange = function(array, a, b) {
    let cp = copy(array);
    let temp = cp[a];
    cp[a] = cp[b];
    cp[b] = temp;
    return cp;
};

this.transform = function(/*[]*/ seq, /*{}*/ prevEcx, /*number*/ deepness){
    //масив з результатами обміну
    let allExchanges = [];

    let b = 0; //куди
    let a = 0; //звідки
    for(let i = 0; i < maxPossibleWays; i++){
        b++;
        if( b === length){
            a++;
            b = a+1;
        }

        let weight = getWeight(this.getDerivs(this.exchange(seq, a, b)));

        let exc = {'from':a, 'to':b};
```

```
            allExchanges[i] = {"weight":weight, "exchange":exc};
        }

        allExchanges.sort(compare);

        if(allExchanges[0].weight[0] === size*size){
            console.log("Знайдена відповідь на глибині " + (deepness+1));
            let res = [];
            res.push(allExchanges[0].exchange);
            return res;
        }

        for(let i = 0; i < allExchanges.length; i++){

            if(compare(allExchanges[i], prevEcx) >= 0){
                return null;
            }

            let e = allExchanges[i].exchange;
            let newSeq = this.exchange(seq, e.from, e.to);
            let goInDeep = this.transform(newSeq, allExchanges[i], deepness+1);

            if(goInDeep == null)
                continue;

            res = [];
            res.push(e);
            for(let i = 0; i < goInDeep.length; i++)
                res.push(goInDeep[i]);
            return res;
        }



        return null;

    };

    function compare(ww1, ww2){
        let w1 = ww2.weight;
        let w2 = ww1.weight;
        let res = 0;

        for(let i = 0; i < w1.length; i++){
            res = w1[i]-w2[i];
            if(res !== 0)
                return res;
        }
        return 0;
    }
}
```